1.2 PROPOSITIONAL LOGIC

The propositional logic represents logic through *propositions* and *logical connectives*. We may define *proposition* as an elementary atomic sentence that may take either ***true*** value or ***false*** value but may not take any other value.

Consider the following examples :

It is raining.	[It is a proposition as it may either be *true* or *false*.]
Australia won the ICC World Cup 2007.	[It is also a proposition as it is *true*]
India is a continent.	[It is a proposition as it is *false*]
What did you eat ?	[It is not a proposition as it does not result in *true* or *false*.]
How are you ?	[Not a proposition for the similar reason as above]

ISC XII

Definition

A Proposition is an elementary atomic sentence that may either be true or false but may take no other value.

Propositions are also called *sentences* or *statements*. After this introduction, let us now talk about terms and symbols used in propositional logic.

1.2.1 Terms and Symbols

A ***simple proposition*** is one that does not contain any other proposition as a part. We will use the lower-case letters, $p, q, r, \ldots$, as symbols for simple statements or propositions.

A ***compound proposition*** is one with two or more simple propositions as parts or what we will call ***components.*** A component of a compound is any whole proposition that is part of a larger proposition ; components may themselves be compounds.

For example, following are compound propositions :

It is raining *and* wind is blowing.
Take it *or* leave it.
If you work hard *then* you will be rewarded.

An **operator** (or ***connective***) joins simple propositions into compounds, and joins compounds into larger compounds. We will use the symbols, +, ., ⇒, and ⇔ to designate the sentential connectives. They are called *sentential* connectives because they join *sentences* (or what we are calling *statements* or *propositions*). *The symbol, ~, is the only operator that is not a connective* ; it affects single statements only, and does not join statements into compounds.

The symbols for statements and for operators comprise our notation or symbolic language. Parentheses serve as punctuation.

Different types of **connectives** (or operators) used in propositional logic are as given below :

1. **Disjunctive** (Also called **OR**). Represented by symbols + or ∨. Disjunction means one of the two arguments is *true* or both *e.g.*, $p + q$ (or $p \vee q$) means ***p* OR *q***. It's meaning is **either *p* is true, or *q* is true, or both.**

2. **Conjunctive** (Also called **AND**). Represented by symbols **.** or **&** or ∧. Conjunction means both arguments are true *e.g.*, $p \cdot q$ (or $p \,\&\, q$) means ***p* AND *q***. Its meaning is **both *p* and *q* are true.**

A compound statement is ***truth-functional*** if its truth value as a whole can be figured out solely on the basis of the truth values of its parts or components. A connective is truth-functional if it makes only compounds that are truth-functional. For example, if we knew the truth values of p and of q, then we could figure out the truth value of the compound, $p + q$. Therefore the compound, $p + q$, is a truth-functional compound and disjunction is a truth-functional connective.

Definition

A **Truth Table** is a complete list of possible truth values of a proposition.

All four of the connectives we are studying (disjunction, conjunction, implication, and equivalence) are truth-functional. Negation is a truth-functional operator. With these four connectives and negation we can express ***all*** the truth-functional relations among statements. A truth table helps us express it.

ISC XII

Let us now learn to make truth tables for all the connectives, we have learnt so far.

(*i*) Negation (NOT). The NOT operator works on single proposition, thus, it is also called *unary connective* sometimes. If p denotes a proposition, then its negation will be denoted by $\sim p$ or p' or $\bar{p}$. If p is 0 (false), then $\sim p$ is 1 (true) and if p is 1 (true) then $\sim p$ is 0 (false). The truth table for this operation is shown as follows :

Table 1.1 *Truth table for Negation (NOT)*

p	$\bar{p}$
0	1
1	0

Also note that

NOT (NOT p) results into p itself *i.e.*,

$$\bar{\bar{p}} = p$$

or $$(p')' = p$$

or $$\sim(\sim p) = p.$$

(*ii*) Disjunction (OR). The OR connective works with more than one proposition. The compound $p + q$ has two (2) component propositions (p and q), each of which can be true or false. So there are four (2^2) possible combinations. The disjunction of p with q (denoted as $p + q$ or $p \vee q$) will be true whenever p is true or q is true or both are true. Consider the truth table given below :

Table 1.2. Truth table for Disjunction (OR)

p	q	$p + q$
0	0	0
0	1	1
1	0	1
1	1	1

Note

If a compound has n distinct components, there will be 2^n rows in its truth table.

(iii) Conjunction (AND). The AND connective also works with more than one proposition. The compound $p \cdot q$ (or p & q) will be *true* whenever both p and q are true.

Table 1.3 *Truth table for Conjunction (AND)*

p	q	p.q
0	0	0
0	1	0
1	0	0
1	1	1

UNIT I

(iv) Implication (If.. Then / Conditional). In the conditional $p \Rightarrow q$, the first proposition (the if-clause) p here, is called the **antecedent** and the second proposition (then clause) q here, is called the **consequent**. In more complex *conditionals*, the *antecedent* and the *consequent* could themselves be *compound propositions*. The conditional $p \Rightarrow q$ will be *false* when p is *true* and q is *false*. *For* all other input combinations, it will be *true*.

Table 1.4 *Truth table for If.. Then*

p	q	$p \Rightarrow q$
0	0	1
0	1	1
1	0	0
1	1	1

The conditional $p \Rightarrow q$ may be expressed as follows :

$$p \Rightarrow q = p' + q$$

(v) Equivalence (If and only If / Bi-conditional). A bi-conditional results into *false* when one of its component proposition is *true* and the other is *false*. That is, $p \Leftrightarrow q$ will be 0 (false) when p *is 0 and* q is 1 Or p is 1 and q is 0. For all other inputs, $p \Leftrightarrow q$ is 1.

Table 1.5 *Truth table for If and only if.*

p	q	$p \Leftrightarrow q$
0	0	1
0	1	0
1	0	0
1	1	1

The bi-conditional $p \Leftrightarrow q$ may also be expressed as :

$$p \Leftrightarrow q = pq + p' \cdot q'$$

Some Related Terms

Contingencies The propositions that have some combination of 1's and 0's in their truth table column, are called *contingencies.*

Tautologies The propositions having nothing but 1's in their truth table column, are called *tautologies.*

Contradictions The propositions having nothing but 0's in their truth table column, are called *contradictions.*

ISC XII

Consistent Statements Two statements are *consistent* if and only if their conjunction is *not a contradiction.*

Converse The converse of a conditional proposition is determined by interchanging the antecedent and consequent of given conditional. It results into a new conditional. *e.g.,* Converse of $p \Rightarrow q$ is $q \Rightarrow p$.

That is, if

p : It is raining.
q : Sky is not clear.
then, $p \Rightarrow q$ = *If it is raining then sky is not clear.*

It's converse will be new conditional as given below :

$q \Rightarrow p$ = *If sky is not clear then it is raining.*

Inverse The inverse of a conditional proposition is another conditional having negated *antecedent* and *consequent*. That is, the inverse of $p \Rightarrow q$ is $p' \Rightarrow q'$. *e.g.,* if

p : It is raining.
q : Sky is not clear.
then, $p \Rightarrow q$ = If it is raining then sky is not clear.

It's inverse will be a new conditional as given below :

$p' \Rightarrow q'$ = If it is not raining then sky is clear.

Contrapositive The contrapositive of a conditional is formed by creating another conditional that takes its *antecedent* as *negated consequent* of earlier conditional and *consequent* as *negated antecedent* of earlier conditional. That is, contrapositive of $p \Rightarrow q$ is $\sim q \Rightarrow \sim p$ or $\bar{q} \Rightarrow \bar{p}$ or $q' \Rightarrow p'$

1.2.3 Some Equivalence Propositional Laws

Two sentences are **equivalent** if they have the same truth value under every interpretation *i.e.,* both the sentences possess the same *truth set.*

In the following lines, we are giving some equivalence laws used in propositional logic. We are giving them without proofs, since their proofs are beyond the scope of this book.

Table 1.6 *Some Equivalence Laws*

1.	$0 + p = p$ $0 . p = 0$	Properties of 0
2.	$1 + p = 1$ $1 . p = p$	Properties of 1
3.	$p + pq = p$ $p + (p + q) = p$	Absorption law
4.	$\bar{\bar{p}} = p$	Involution
5.	$p + p = p$ $p . p = p$	Idempotence law
6.	$p + \bar{p} = 1$ $p . \bar{p} = 0$	Complementarity law
7.	$p + q = q + p$ $p . q = q . p$	Commutative law
8.	$(p + q) + r = p + (q + r)$ $(p . q) . r = p .(q . r)$	Associative law
9.	$p .(q + r) = (p . q) + (p . r)$ $p + (q . r) = (p + q) .(p + r)$ $p + \bar{p} q = p + q$	Distributive law
10.	$\overline{p + q} = \bar{p} . \bar{q}$ $\overline{p . q} = \bar{p} + \bar{q}$	De Morgan's law
11.	$p \Rightarrow q = \bar{p} + q$	Conditional elimination
12.	$p \Leftrightarrow q = (p \Rightarrow q) . (q \Rightarrow p)$	Bi-conditional elimination

Let us now have a look at some examples.

Example 1.1. *Construct a truth table for the expression* $(A.(A + B))$. *What single term is the expression equivalent to ?*

Solution.

A	**B**	**A + B**	**(A · (A + B))**
0	0	0	0
0	1	1	0
1	0	1	1
1	1	1	1

Looking at the table, we find that columns **(*A* . (*A* + B))** and **A** are identical. That is, they possess the same *truth set*. Hence the given expression **(*A*. (*A* + B))** is equivalent to *A*.

Example 1.2. *Using truth table, prove that $p \Rightarrow q$ is equivalent to $\sim q \Rightarrow \sim p$.*

Solution.

p	q	$\sim q$	$\sim p$	$p \Rightarrow q$	$\sim q \Rightarrow \sim p$
0	0	1	1	1	1
0	1	0	1	1	1
1	0	1	0	0	0
1	1	0	0	1	1

From the above truth table it is obvious that columns $p \Rightarrow q$ and $\sim q \Rightarrow \sim p$ are identical i.e. possessing same truth set (1, 1, 0, 1). Hence it is proved that

$$p \Rightarrow q = \sim q \Rightarrow \sim q.$$

This rule is also called *transposition*.

Example 1.3. *Prove that $p \Rightarrow q = \overline{p} + q$.*

Solution.

p	q	$\overline{p}$	$p \Rightarrow q$	$\overline{p} + q$
0	0	1	1	1
0	1	1	1	1
1	0	0	0	0
1	1	0	1	1

From the above truth table, we find that columns $p \Rightarrow q$ and $\overline{p} + q$ are possessing same truth set (1, 1, 0, 1). Hence proved that

$$p \Rightarrow q \;=\; \overline{p} + q.$$

Example 1.4. *Prove that $p \Leftrightarrow q = q \Leftrightarrow p$.*

Solution.

p	q	$p \Leftrightarrow q$	$q \Leftrightarrow p$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	1	1

From the above truth table, we find that both propositions $p \Leftrightarrow q$ and $q \Leftrightarrow p$, possess same truth set (1, 0, 0, 1). Hence proved that

$$p \Leftrightarrow q \;=\; q \Leftrightarrow p.$$

Example 1.5. *Prove that $p \Leftrightarrow q \;=\; (p \Rightarrow q).(q \Rightarrow p)$.*

Solution.

p	q	$p \Leftrightarrow q$	$p \Rightarrow q$	$q \Rightarrow p$	$(p \Rightarrow q).(q \Rightarrow p)$
0	0	1	1	1	1
0	1	0	1	0	0
1	0	0	0	1	0
1	1	1	1	1	1

Since the columns $p \Leftrightarrow q$ and $(p \Rightarrow q).(q \Rightarrow p)$ are identical, it is proved that

$$p \Leftrightarrow q = (p \Rightarrow q).(q \Rightarrow p)$$

Example 1.6. *Consider some simple propositions given below :*

A : It is raining.

B : Wind is blowing.

C : I am not driving.

From these, create the following compound proportions.

(*i*) $A \vee B$ (*ii*) $\sim B$ (*iii*) $\sim B.C$ (*iv*) $A.\sim C$ (*v*) $A + B.C.$

Solution.

(*i*) $A \vee B$: It is raining OR wind is blowing.

(*ii*) $\sim B$: Wind is NOT blowing.

(*iii*) $\sim A.C$: It is NOT raining AND I am not driving.

(*iv*) $A.\sim C$: It is raining AND I am driving.

(*v*) $A + B.C$: It is raining OR wind is blowing AND I am not driving.

Example 1.7. *Prove that $X + 1$ is a tautology.*

Solution.

X	1	X + 1
0	1	1
1	1	1

Since the column $X + 1$ has all trues (1's) in its column, it is a tautology.

Example 1.8. *Prove that $X + X'$ is a tautology and $X.X'$ is a contradiction.*

Solution.

X	X'	X + X'	X . X'
0	1	1	0
1	0	1	0

$X + X'$ has all 1's in its truth set, hence it is a tautology.

$X.X'$ has all 0's in its truth set, hence it is a contradiction.

1.2.4 Drawing Conclusions – Syllogism

While studying logic, many a times conclusions are drawn from given two or more logic statements. This process, rather logical process of drawing conclusions from given logic statements, is called ***syllogism***. The given statements or propositions are called ***premises***.

Definition

The logical process of drawing conclusions from given propositions is called **syllogism.** The propositions used to draw conclusion are called **premises.**

To draw conclusions, we may use any of the two methods available for it :

- Truth Table Method
- Algebraic Method

1. Truth Table Method

In this method, a truth table (TT) is drawn for all the *given premises* and the *conclusion to be* drawn. Then a conditional is prepared having the *antecedent as conjunction of all given premises* and *consequent* as the *conclusion* to be drawn. If this conditional results into a tautology (all true's *i.e.*, 1's) then the given conclusion is established. For this consider the example given below :

Example 1.9. *From premises p and $p \Rightarrow q$, infer q.*

Solution. Given premises are :

(P1) : p

(P2) : $p \Rightarrow q$

Conclusion to be drawn C : q

As per the TT method, we have to prepare a truth table for given premises p, $p \Rightarrow q$, the conjunction of these given premises *i.e.*, $p.(p \Rightarrow q)$ and the conclusion to be drawn *i.e.*, q.

ISC XII

Also there should be a column for

$P1 . P2 \Rightarrow C$ *i.e.*, $[p . (p \Rightarrow q)] \Rightarrow q$

Our truth table will look like :

p	q	p ⇒ q	p . (p ⇒ q)	[p . (p ⇒ q)] ⇒ q
0	0	1	0	1
0	1	1	0	1
1	0	0	0	1
1	1	1	1	1

From the above truth table, it is clear that the conditional having *antecedent* as the conjunction of premises $[p.(p + q)]$ and *consequent* as the conclusion (q), is a tautology. Hence it is established that from given premises p and $p \Rightarrow q$ conclusion drawn is q. That is

$$\begin{array}{l} p \\ p \Rightarrow q \\ \hline q \end{array}$$

Example 1.10. From $p \Rightarrow q$ and $q \Rightarrow r$, infer $p \Rightarrow r$.

Solution. Given premises are :

(P1) : $p \Rightarrow q$

(P2) : $q \Rightarrow r$

Conclusion (C) to be drawn : $p \Rightarrow r$ as per TT method.

We shall prepare a truth table having column for

P1 (*i.e.*, $p \Rightarrow q$), P2 (*i.e.*, $q \Rightarrow r$), P1 . P2 *i.e.*, $(p \Rightarrow q).(q \Rightarrow r)$,

C (*i.e.*, $p \Rightarrow r$) and P1 . P2 ⇒ C *i.e.*, $[(p \Rightarrow q).(q \Rightarrow r)] \Rightarrow (p \Rightarrow r)$

p	q	r	p ⇒ q	q ⇒ r	(p ⇒ q) . (q ⇒ r)	p ⇒ r	[(p ⇒ q) . (q ⇒ r)] ⇒ (p ⇒ r)
0	0	0	1	1	1	1	1
0	0	1	1	1	1	1	1
0	1	0	1	0	0	1	1
0	1	1	1	1	1	1	1
1	0	0	0	1	0	0	1
1	0	1	0	1	0	1	1
1	1	0	1	0	0	0	1
1	1	1	1	1	1	1	1

From the table, we derive that

$P1 . P2 \Rightarrow C$ is a tautology *i.e.*, having all 1's in its truth set. Hence concluded that

$$\begin{array}{c} p \Rightarrow q \\ q \Rightarrow r \\ \hline p \Rightarrow r \end{array}$$

(ii) Algebraic Method

In this method, to draw a conclusion from given premises, **conditional elimination** is carried out. That is, in place of a conditional $p \Rightarrow q$ its equivalent $\bar{p} + q\,(\sim p + q)$ is substituted and then it is checked whether the conditional having *antecedent* as the *conjunction-of-all premises* and *consequent* as the *conclusion-to-be-drawn*, is a tautology or not. To understand this, consider the example given below.

Example 1.11. *From p and $p \Rightarrow q$, infer q.*

Solution. Given premises are

(P1): $\quad p$

(P2): $\quad p \Rightarrow q$

Conclusion (C) to be drawn : q.

Let us compute $\quad P1 . P2 \Rightarrow C$

i.e., $\quad [p.(p \Rightarrow q)] \Rightarrow q$

Carrying out conditional elimination *i.e.*, substituting $p \Rightarrow q$ with $\bar{p} + q$, we get

$$= [p.(\bar{p} + q)] \Rightarrow q$$

$$= (p.\bar{p} + p.q) \Rightarrow q$$

$$= 0 + (p.q) \Rightarrow q \qquad (\because p\bar{p} = 0)$$

Carrying out conditional elimination once again, we get

$$(\overline{p.q}) + q$$

$$= \bar{p} + \bar{q} + q \qquad (\because \overline{pq} = \bar{p} + \bar{q}, \text{ Rule 10, table 1.6})$$

$$= \bar{p} + 1 \qquad (\because \bar{q} + q = 1, \text{ Rule 6, table 1.6})$$

$$= \bar{p} + 1 \qquad (\because \bar{p} + 1 = p, \text{ Rule 2, table 1.6})$$

$$= 1$$

Hence the result is established.

Example 1.12. *From $p \Rightarrow q$ and $q \Rightarrow r$, infer $p \Rightarrow r$.*

Solution. Given premises are : $\quad p \Rightarrow q, \quad q \Rightarrow r$

and conclusion to be drawn is $p \Rightarrow r$.

Thus we have to establish that $\quad [(p \Rightarrow q).(q \Rightarrow r)] \Rightarrow (p \Rightarrow r)$

Carrying out conditional elimination, we get $\;[(\bar{p} + q).(\bar{q} + r)] \Rightarrow (\bar{p} + r)$

Carrying out conditional elimination once again, we get

$$= (\overline{\bar{p} + q}).(\overline{\bar{q} + r}) + (\bar{p} + r)$$

$$= (\overline{\bar{p}\; + q}) + (\overline{\bar{q}\; + r}) + (\bar{p} + r) \qquad \text{(De Morgan's Law)}$$

$$= (\bar{\bar{p}}.\bar{q}) + (\bar{\bar{q}}.\bar{r}) + (\bar{p} + r) \qquad \text{(De Morgan's Law)}$$

1.3 DEVELOPMENT OF BOOLEAN ALGEBRA

Long ago Aristotle constructed a complete system of formal logic and wrote six famous works on the subject, contributing greatly to the organization of man's reasoning. For centuries afterward, mathematicians kept on trying to solve these logic problems using conventional algebra but only *George Boole* could manipulate these symbols successfully to arrive at a solution with his own mathematical system of logic. Boole's revolutionary paper '*An investigation of the laws of the thought*' was published in 1854 which led to the development of new system, the *algebra of logic*, 'BOOLEAN ALGEBRA'.

Boole's work remained confined to papers only until 1938 when *Claude E. Shannon* wrote a paper titled '*A Symbolic Analysis of Relay Switching Circuits*'. In this paper he applied Boolean Algebra to solve relay logic problems. As logic problems are binary decisions and boolean algebra effectively deals with these binary values. Thus it is also called '*Switching Algebra*'.

1.4 BINARY VALUED QUANTITIES

Everyday we have to make logic decisions: "Should I carry the book or not?", "Should I use calculator or not?", "Should I miss TV Programme or not?". Each of these questions requires a YES or NO answer as there are only these two possible answers.

Therefore, each of the above mentioned is a binary decision. Binary decision making also applies to formal logic. For example, let us consider the following:

1. Indira Gandhi was the only woman Prime Minister of India.
2. 13 – 2 = 11
3. Delhi is the biggest state in India.
4. What do you say ?
5. What did I say yesterday ?

1st and 2nd sentences are TRUE but 3rd is FALSE. 4th and 5th are questions which can not be answered in TRUE and FALSE.

Definition

The decision which results into either YES (TRUE) or NO (FALSE) is called a Binary Decision.

These sentences which can be determined to be true or false are called logical statements or truth functions and the results TRUE or FALSE are called truth values. The truth values are depicted by logical constants TRUE and FALSE or 1 and 0. 1 means TRUE and 0 means FALSE. And the variables which can store these truth values are called logical variables or ***binary valued variables*** as these can store one of the two values TRUE or FALSE.

1.5 LOGICAL OPERATIONS

There are some specific operations that can be applied on truth functions. Before learning about these operations, you must know about mathematical logical functions and logical operators.

1.5.1 Logical Function or Compound Statement

Algebraic variables like *a, b, c* or *x, y, z* etc. are combined with the help of *mathematical operators* like +, −, ×, / to form algebraic expressions *e.g.*,

$$2 \times A + 3 \times B - 6 \times C = (10 \times Z)/2 \times Y \quad i.e., \quad 2A + 3B - 6C = 10Z/2Y$$

Similarly, logic statements or truth functions are combined with the help of *Logical Operators* like AND, OR and NOT to form a *Compound statement or Logical function. e.g.*,

He prefers tea *not* coffee.

He plays guitar *and* she plays sitar.

I watch TV on Sundays *or* I go for swimming.

ISC XII

These logical operators are also used to combine logical variables and logical constants to form *logical expressions e.g.*, assuming *x, y* are logical variables

X NOT *Y* OR *Z*

Y AND *X* OR *Z*

1.5.2 Logical Operators

Before we start discussion about logical operators, let us first understand *what a Truth Table is.*

Definition

> **Truth Table** is a table which represents all the possible values of logical variables / statements along with all the possible results of the given combinations of values.

For example, following logical statements can have only one of the *two* values (TRUE (YES) or FALSE (NO))

1. I want to have tea.
2. Tea is readily available.

Let us represent all the possible combinations of values the statements can have in the tabular form :

I want to have tea	T	T	F	F
Tea is readily available	T	F	T	F
(Result) I'll have tea	T	F	F	F

T represents True
F represents False

Or If we represent first statement as *X* and second statement as *Y* and result as *R* then the above table can also be written as follows :

Table 1.7

X	*Y*	*R*
1	1	1
1	0	0
0	1	0
0	0	0

Definition

> If result of any logical statement or expression is always TRUE or 1, it is called **Tautology** and if the result is always FALSE or 0 it is called **Fallacy**.

1 represents TRUE value and 0 represents FALSE value.

This is a truth table *i.e., table of truth values of truth functions.*

Now let us proceed with our discussion about logical operators *i.e.*,

(*a*) NOT Operator (*b*) OR Operator (*c*) AND Operator

NOT Operator

This operator operates on single variable and operation performed by NOT operator is called *complementation* and the symbol we use for it is $\bar{\ }$ (bar). Thus $\overline{X}$ means complement of X and $\overline{YZ}$ means complement of YZ. As we know, the variables used in boolean equations have a unique characteristic that they may assume only one of two possibie values 0 and 1, where 0 denotes FALSE and 1 denotes TRUE value. Thus the complement operation can be defined quite simply.

$$\overline{0}=1$$
$$\overline{1}=0$$

Table 1.8 *Truth Table for NOT operator*

X	$\overline{X}$
0	1
1	0

Several other symbols *e.g.* ~, ′ are also used for the complementation symbol. If ~ is used then ~ **X** is read is 'negation of *X*' and if symbol ′ is used then **X**′ is read as complement of X.

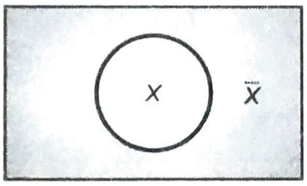


FIGURE 1.1 Venn diagram for $\overline{X}$

NOT operation is singular or unary operation as it operates on single variable.

Venn diagram for $\overline{X}$ is given above where shaded area depicts $\overline{X}$.

OR Operator

A second important operator in boolean algebra is OR operator which denotes operation called *logical addition* and the symbol we use for it is +. The + symbol, therefore, does not have the 'normal' meaning, but is a logical addition or logical OR symbol. Thus $X + Y$ can be read as **X OR Y**. For OR operation the possible input and output combinations are as follows :

$$0+0=0$$
$$0+1=1$$
$$1+0=1$$
$$1+1=1$$

And the truth table of OR operator is given below :

Table 1.9 *Truth Table for OR operator*

X	Y	X + Y
0	0	0
0	1	1
1	0	1
1	1	1

Note that when any one of *X* and *Y* is 1, *X* + *Y* is 1.

It is often convenient to shorten X . Y . Z to XYZ, and using this convention, above expression can be written as $X + \overline{YZ} + \overline{Z}$

To study a boolean expression, it is very useful to construct a table of values for the variables and then to evaluate the expression for each of the possible combinations of variables in turn. Consider the expression $X + \overline{YZ}$. Here three variables X, Y, Z are forming the expression, each of the variables can assume the value 0 or 1. The possible combinations of values may be arranged in ascending order as in Table 1.11

Table 1.11 *Possible Combinations of X, Y and Z*

X	Y	Z
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

Since X, Y, Z are three (3) variables in total. A truth table involving 3 input variables will have 2^3 *i.e.*, 8 rows in total. The left most column will have half of total entries (*i.e.*, 4 entries) as zeros and half as 1's (in total 8). The next column will have no of zero's and 1's halved than first column completing 8 rows and so on. That is why, first column has four 0's and four 1's, next column has two 0's followed by two 1's completing 8 rows in total and the last column has one 0's followed by one 1's completing 8 rows in total.

So a column is added to list Y . Z (Table 1.12)

Table. 1.12 *Truth Table for (Y . Z)*

X	Y	Z	Y . Z
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

AND operation is applied only on columns Y and Z

One more column is now added to list the values of $\overline{YZ}$ (Table 1.13)

Table 1.13 *Truth Table for Y . Z and $\overline{YZ}$.*

X	Y	Z	Y . Z	$\overline{YZ}$
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

Note that $\overline{YZ}$ contains complemented values of YZ

(b) Truth Table for $XY(Z + YZ) + Z$ is as follows :

X	Y	Z	Y	Z	YZ	Z + YZ	XY	XY(Z + YZ)	XY(Z + YZ) + Z
0	0	0	1	1	0	0	0	0	1
0	0	1	1	0	0	1	0	0	0
0	1	0	0	1	1	1	0	0	1
0	1	1	0	0	0	1	0	0	0
1	0	0	1	1	0	0	1	0	1
1	0	1	1	0	0	1	1	1	1
1	1	0	0	1	1	1	0	0	1
1	1	1	0	0	0	1	0	0	0

(c) Truth Table for $A[(B + C) + C]$ is as follows :

A	B	C	B	C	(B + C)	(B + C) + C	A[(B + C) + C]
0	0	0	1	1	1	1	0
0	0	1	1	0	1	1	0
0	1	0	0	1	0	1	0
0	1	1	0	0	1	1	0
1	0	0	1	1	1	1	1
1	0	1	1	0	1	1	1
1	1	0	0	1	0	1	1
1	1	1	0	0	1	1	1

1.6 BASIC LOGIC GATES

After *Shannon* applied boolean algebra in telephone switching circuits, engineers realized that boolean algebra could be applied to computer electronics as well.

In the computers, these boolean operations are performed by logic gates.

What is a Logic Gate ?

Gates are digital (two-state) circuits because the input and output signals are either low voltage (denotes 0) or high voltage (denotes 1). Gates are often called *logic circuits* because they can be analyzed with boolean algebra.

Definition

A Gate is simply an electronic circuit which operates on one or more signals to produce an output signal.

There are *three* types of logic gates :

(i) Inverter (NOT gate)
(ii) OR gate
(iii) AND gate

1.6.1 Inverter (NOT Gate)

An inverter is also called a NOT gate because the output is not the same as the input. The output is sometimes called the *complement* (opposite) of the input. Following tables summarise the operation :

Definition

An **Inverter (NOT Gate)** is a gate with only one input signal and one output signal; the output state is always the opposite of the input state.

Table 1.15 *Truth Table for NOT gate*

X	$\overline{X}$
Low	High
High	Low

Table 1.16 *Alternative truth table for NOT gate*

X	$\overline{X}$
0	1
1	0

A low input *i.e.*, 0 produces high output *i.e.*, 1, and vice versa. The symbol for inverter is given in adjacent Fig. 1.4.

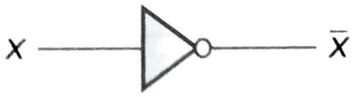


FIGURE 1.4 NOT gate symbol

1.6.2 OR Gate

If all inputs are 0 then output is also 0. If one or more inputs are 1, the output is 1.

An OR gate can have as many inputs as desired. No matter how many inputs are there, the action of OR gate is the same : one or more 1 (high) inputs produce output as 1.

Definition

The **OR Gate** has two or more input signals but only one output signal. If any of the input signals is 1 (high), the output signal is 1 (high).

Following tables show OR action

Table 1.17 *Two input OR gate*

X	Y	F
0	0	0
0	1	1
1	0	1
1	1	1

$F = X + Y$

Table 1.18 *Three input OR gate*

X	Y	Z	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

$F = X + Y + Z$

The symbol for OR gate is given below :

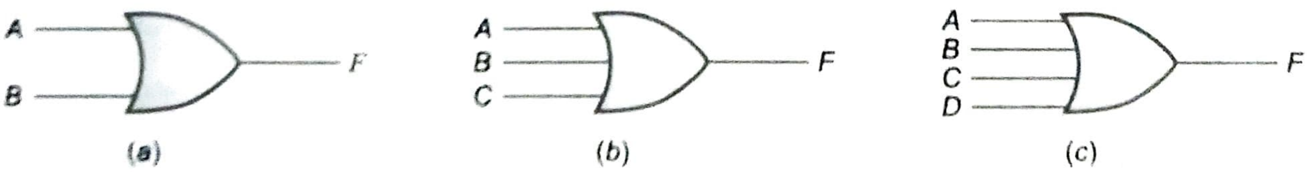


FIGURE 1.5 (a) Two input OR gate (b) Three input OR gate (c) Four input OR gate.

1.6.3 AND gate

> **Definition**
> The **AND Gate** can have tw[o] or more than two input sign[als] and produce an output sign[al]. When **all the** inputs are 1 i.e. high **then** the output is [1] otherwise output is 0 only.

If any of the inputs is 0, the output is 0. To obtain output as 1, all inputs must be 1.

An AND gate can have as many inputs as desired. Following tables illustrate AND action.

Table 1.19 *Two input AND gate*

X	Y	F
0	0	0
0	1	0
1	0	0
1	1	1

Here, $F = X.Y$

Table 1.20 *Three input AND gate*

X	Y	Z	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Here, $F = X.$

The symbol for AND is

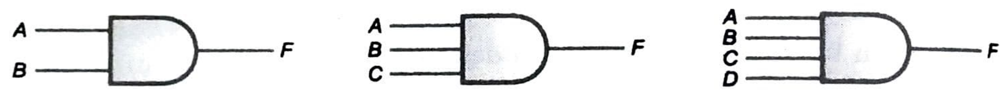


FIGURE 1.6 (a) 2-input AND gate (b) 3-input AND gate (c) 4-input AND gate

1.7 BASIC POSTULATES OF BOOLEAN ALGEBRA

Boolean algebra, being a system of mathematics, consists of *fundamental laws* that are us[ed to] build a workable, cohesive framework upon which are based the theorems of boolean alg[ebra]. These fundamental laws are known as *Basic postulates of boolean algebra.* These postulates state relations in boolean algebra, that follow :

I. If $X \neq 0$ then $X = 1$; and If $X \neq 1$ then $X = 0$

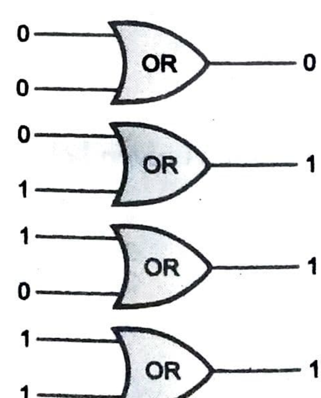
II. OR Relations (*Logical Addition*)

$0 + 0 = 0$

$0 + 1 = 1$

$1 + 0 = 1$

$1 + 1 = 1$



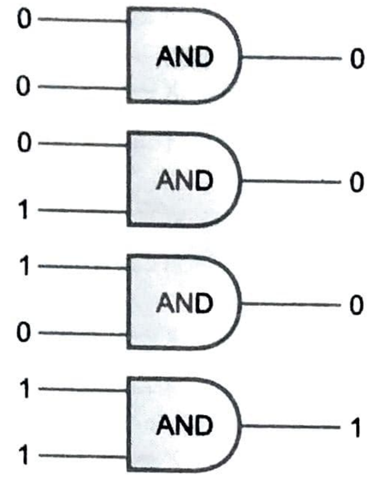
III. AND Relations (Logical Multiplication)

$0 . 0 = 0$

$0 . 1 = 0$

$1 . 0 = 0$

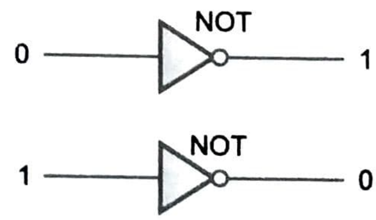
$1 . 1 = 1$



IV. Complement Rules

$\bar{0} = 1$

$\bar{1} = 0$



1.8 PRINCIPLE OF DUALITY

This is a very important principle used in boolean algebra. This states that *starting with a boolean relation, another boolean relation can be derived by*

1. Changing each OR sign (+) to an AND sign (.)
2. Changing each AND sign (.) to an OR sign (+)
3. Replacing each 0 by 1 and each 1 by 0.

The derived relation using duality principle is called *dual of original expression.*

For instance, we take *postulate II* related to logical addition, which states

(*a*) $0 + 0 = 0$ (*b*) $0 + 1 = 1$ (*c*) $1 + 0 = 1$ (*d*) $1 + 1 = 1$

Now working according to above guidelines, + is changed to . and 0's are replaced by 1's, these become

(*i*) $1.1 = 1$ (*ii*) $1.0 = 0$ (*iii*) $0.1 = 0$ (*iv*) $0.0 = 0$

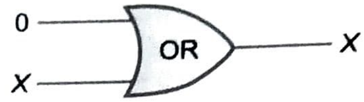
which are nothing but *same as that of postulate III* related to logical multiplication. So *i, ii, iii, iv* are the duals of *a, b, c* & *d*. We'll be applying this duality principle in the theorems of boolean algebra which is our next topic.

1.9 BASIC THEOREMS OF BOOLEAN ALGEBRA

Basic postulates of boolean algebra are used to define *basic theorems of boolean algebra* that provide all the tools necessary for manipulating boolean expressions. Although simple in appearance, these theorems may be used to construct the boolean algebra.

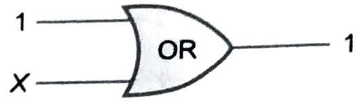
1.9.1 Properties of 0 and 1

(a) $0 + X = X$



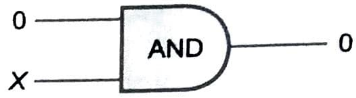
(gate representation of (a))

(b) $1 + X = 1$



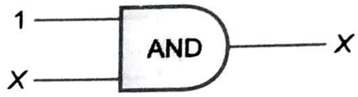
(gate representation of (b))

(c) $0 . X = 0$



(gate representation of (c))

(d) $1 . X = X$



(gate representation of (d))

Proof.

(a) $0 + X = X$

Truth table for above expression is given below in Table 1.21, where R signifies the outp

Table 1.21 *Truth Table for* $0 + X = X$.

0	X	R
0	0	0
0	1	1

as X can have values either 0 or 1 (*postulate I*) both the values ORed with 0 produce the same ou *as that of* X. Hence proved

(b) $1 + X = 1$

Truth table for this expression is given below in Table 1.22, where R signifies the outp

Table 1.22 *Truth table for* $1 + X = 1$

1	X	R
1	0	1
1	1	1

Again X can have values 0 or 1. Both the values (0 and 1) ORed with 1 produce the outpu 1. Hence proved. Therefore $1 + X = 1$ *is* ***a tautology***.

(c) $0 . X = 0$

As both the possible values of X (0 and 1) are to be ANDed with 0, so, the truth table for expression is as follows where (R signifies the output)

Table 1.23 *Truth Table for* $0 . X = 0$.

0	X	R
0	0	0
0	1	0

Both the values of X(0 and 1) when ANDed with produce the *output as* 0. Hence prov Therefore, $0 . X = 0$ is a ***fallacy***.

(d) $1 . X = X$

Now both the possible values of X (0 and 1) are to be ANDed with 1. Thus the truth table for it will be as follows :

Table 1.24 *Truth Table for* $1 . X = X$

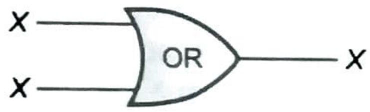
1	X	R
1	0	0
1	1	1

Now observe both the values (0 and 1) when ANDed with 1 produce the *same output as that of* X. Hence proved. Here *b* and *c* are duals of each other and *a* and *d* are duals of each other.

1.9.2 Indempotence Law

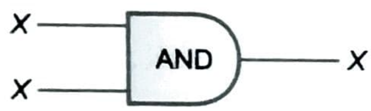
This law states that

(a) $X + X = X$ *i.e.,*



(*gate representation for (a)*)

and (b) $X . X = X$ *i.e.,*



(*gate representation of (b)*)

Proof.

(a) $X + X = X$

To prove this law, we will make truth table for above expression. As X is to be ORed with itself only, we will prepare truth table with the two possible values of X (*i.e.,* 0 and 1).

Table 1.25 *Truth Table for* $X + X = X$

X	X	R
0	0	0
1	1	1

$0 + 0 = 0$ (*ref. postulate II*)

and $1 + 1 = 1$ (*ref. postulate II*)

$\Rightarrow$ $X + X = X$, as it holds true for both values of X. Hence proved.

(b) $X . X = X$

Here X is ANDed with itself. Again we will prepare truth table for this expression taking 2 possible values of X (0 and 1)

Table 1.26 *Truth Table for* $X . X = X$

X	X	R
0	0	0
1	1	1

$0 . 0 = 0$ (*ref. postulate III*)

and $1 . 1 = 1$ (*ref. postulate III*)

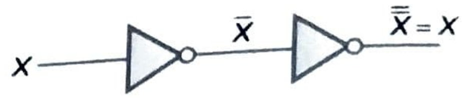
$\Rightarrow$ $X . X = X$, as it holds true for both values of X. Hence proved.

(a) and (b) are duals of each other.

1.9.3 Involution

This law states that

$(\overline{\overline{X}}) = X$ *i.e.,*



To prove this, again we'll prepare truth table which is given below.

Table 1.27 *Truth Table for* $\overline{\overline{X}} = X$

X	$\overline{X}$	$\overline{\overline{X}}$
0	1	0
1	0	1

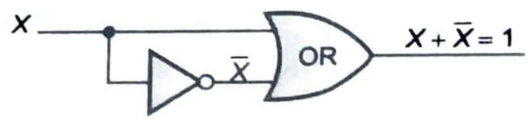
First column represents possible values of X, second column represents complement of X (*i.e.,* $\overline{X}$) and the third column represents complement of $\overline{X}$ (*i.e.,* $\overline{\overline{X}}$) which is same as that of X. Hence proved.

This law is also called *double-inversion rule.*

1.9.4 Complementarity Law

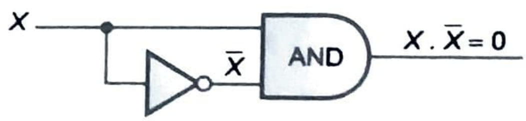
These laws state that

(*a*) $X + \overline{X} = 1$



(*gate representation of* (*a*))

and (*b*) $X . \overline{X} = 0$



(*gate representation of* (*b*))

Proof.

(*a*) We will prove $X + \overline{X} = 1$ with the help of truth table which is given below :

Table 1.28 *Truth Table for* $X + \overline{X} = 1$

X	$\overline{X}$	$X + \overline{X}$
0	1	1
1	0	1

Here, in the first column possible values of X have been taken, second column consists of $\overline{X}$ values (*complement values of* X), X and $\overline{X}$ values are ORed and the output is shown in third column as

$$0 + 1 = 1, \quad \text{(ref. postulate II)}$$
$$1 + 0 = 1 \quad \text{(ref. postulate II)}$$

$\Rightarrow$ $X + \overline{X} = 1$, as it holds true for both possible values of X. Hence proved. It is a *tautology* (*b*) $X \cdot \overline{X} = 0$

Truth table for this expression is as follows :

Table 1.29 *Truth table for* $X \cdot \overline{X} = 0$

X	$\overline{X}$	$X \cdot \overline{X}$
0	1	0
1	0	0

as $\qquad 0.1 = 0$ (*ref. postulate III*)

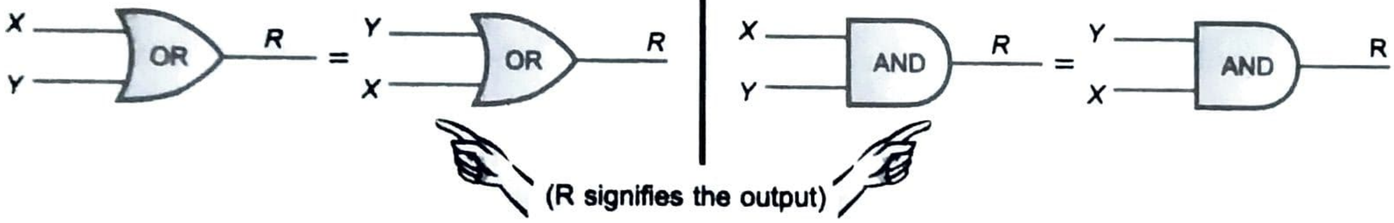
and $\qquad 1.0 = 0$ (*ref. postulate III*)

$\Rightarrow X \cdot \overline{X} = 0$, as it holds true for both the values of X. Hence proved. *Observe here* $X \cdot \overline{X} = 0$ *is dual* of $X + \overline{X} = 1$. Changing (+) to (.) and 1 to 0, and we get $X \cdot \overline{X} = 0$. It is a ***fallacy***.

1.9.5 Commutative law

These laws state that

(*a*) $X + Y = Y + X$ $\qquad$ (*b*) $X \cdot Y = Y \cdot X$



Proof. (*a*) Truth Table for $X + Y = Y + X$ is given below :

Table 1.30 *Truth Table for* $X + Y = Y + X$

X	Y	$X + Y$	$Y + X$
0	0	0	0
0	1	1	1
1	0	1	1
1	1	1	1

Compare the columns $X + Y$ and $Y + X$, both of these are identical. Hence proved.

(*b*) Truth table for $X \cdot Y = Y \cdot X$ is given below :

Table 1.31 *Truth table for* $X \cdot Y = Y \cdot X$

X	Y	$X \cdot Y$	$Y \cdot X$
0	0	0	0
0	1	0	0
1	0	0	0
1	1	1	1

Both of the columns $X \cdot Y$ and $Y \cdot X$ are identical, hence proved.

1.9.6 Associative law

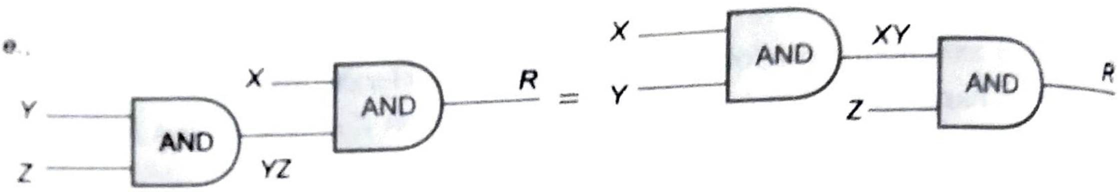
These laws state that

(*a*) $X + (Y + Z) = (X + Y) + Z$

i.e.

(*b*) $X (YZ) = (XY) Z$

i.e.,



Proof. (*a*) Truth table for $X + (Y + Z) = (X + Y) + Z$ is given below :

Table 1.32 *Truth Table for* $X + (Y + Z) = (X + Y) + Z$

X	Y	Z	Y + Z	X + Y	X + (Y + Z)	(X + Y) + Z
0	0	0	0	0	0	0
0	0	1	1	0	1	1
0	1	0	1	1	1	1
0	1	1	1	1	1	1
1	0	0	0	1	1	1
1	0	1	1	1	1	1
1	1	0	1	1	1	1
1	1	1	1	1	1	1

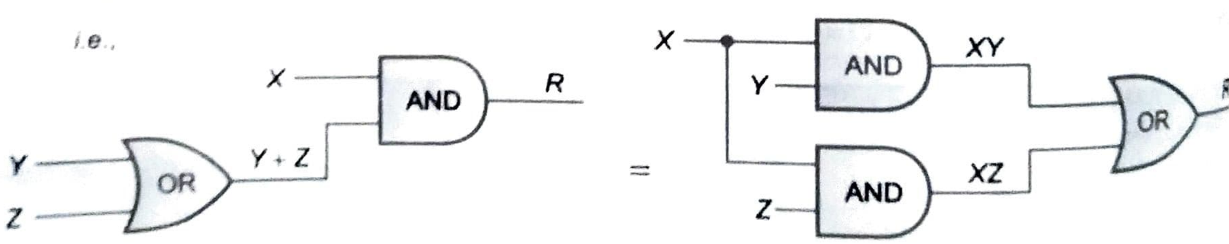
Compare the columns $X + (Y + Z)$ and $(X + Y) + Z$, both of these are identical. Hence prove
Since rule (*b*) is dual of rule (*a*), hence it is also proved.

1.9.7 Distributive law

This law states that

(*a*) $X (Y + Z) = XY + XZ$

i.e.,

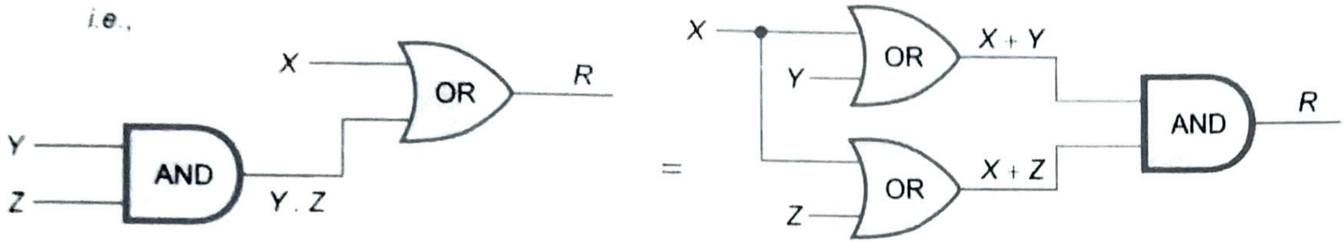


---

1. If a boolean expression is true then its dual is also true.

(*b*) **X + YZ = (X + Y)(X + Z)**

i.e.,



Proof.

(*a*) Truth table for $X(Y + Z) = XY + XZ$ is given below :

Table 1.33 *Truth Table for* $X(Y + Z) = XY + XZ$

X	Y	Z	Y + Z	XY	XZ	X(Y + Z)	XY + XZ
0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	1	0	0	0	0
0	1	1	1	0	0	0	0
1	0	0	0	0	0	0	0
1	0	1	1	0	1	1	1
1	1	0	1	1	0	1	1
1	1	1	1	1	1	1	1

Both the columns $X(Y + Z)$ and $XY + YZ$ are identical, hence proved.

(*b*) Since rule (*b*) is dual of rule (*a*), hence it is also proved.

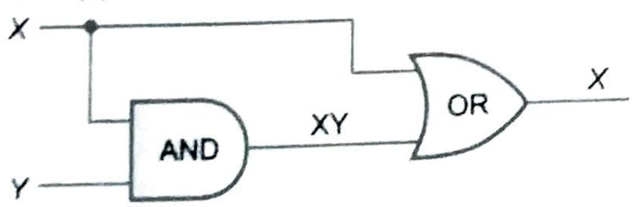
However, we are giving the algebraic proof of law $X + YZ = (X + Y)(X + Z)$

$$\begin{aligned}
\text{R.H.S.} &= (X + Y)(X + Z) = XX + XZ + XY + YZ \\
&= X + XZ + XY + YZ && (XX = X \textit{ Indempotence law}) \\
&= X + XY + XZ + YZ = X(1 + Y) + Z(X + Y) \\
&= X.1 + Z(X + Y) && (1 + Y = 1 \textit{ property of 0 and 1}) \\
&= X + XZ + YZ && (X.1 = X \textit{ property of 0 and 1}) \\
&= X(1 + Z) + YZ \\
&= X.1 + YZ && (1 + Z = 1 \textit{ property of 0 and 1}) \\
&= X + YZ && (X.1 = X \textit{ property of 0 and 1}) \\
&= \text{L.H.S.} \quad \text{Hence proved}
\end{aligned}$$

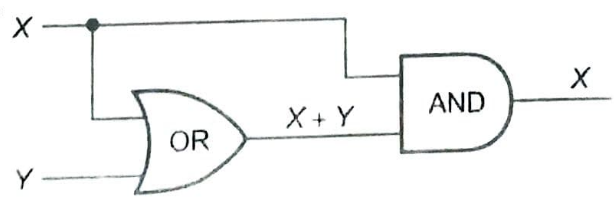
1.9.8 Absorption law

According to this law

i.e., (*a*) **X + XY = X**



i.e., (*b*) **X (X + Y) = X**



$= X + XY + \overline{X}Y$

$= X + Y(X + \overline{X})$

$= X + Y.1$ ($X + \overline{X} = 1$ *complementarity law*)

$= X + Y$ ($Y.1 = Y$ *property of 0 and 1*)

= R.H.S. Hence proved.

All the theorems of boolean algebra, which we have covered so far, are summarised in the following table :

Table 1.35 *Boolean Algebra Rules*

1.	$0 + X = X$	*Properties of 0*
2.	$0 . X = 0$	
3.	$1 + X = 1$	*Properties of 1*
4.	$1 . X = X$	
5.	$X + X = X$	*Indempotence law*
6.	$X . X = X$	
7.	$\overline{\overline{X}} = X$	*Involution*
8.	$X + \overline{X} = 1$	*Complementarity law*
9.	$X . \overline{X} = 0$	
10.	$X + Y = Y + X$	*Commutative law*
11.	$X . Y = Y.X$	
12.	$X + (Y + Z) = (X + Y) + Z$	*Associative law*
13.	$X (YZ) = (XY) Z$	
14.	$X (Y + Z) = XY + XZ$	*Distributive law*
15.	$X + YZ = (X + Y)(X + Z)$	
16.	$X + XY = X$	*Absorption law*
17.	$X * (X + Y) = X$	
18.	$X + \overline{X}Y = X + Y$	

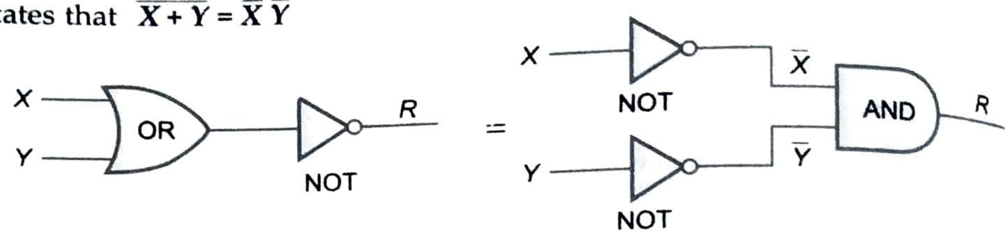
1.10 DEMORGAN'S THEOREMS

One of the most powerful identities used in boolean algebra is DeMorgan's theorem. *Augustus DeMorgan* had paved the way to boolean algebra by discovering these *two* important theorems. This section introduces these *two* theorems of DeMorgan.

UNIT I

1.10.1 DeMorgan's First Theorem

It states that $\overline{X+Y} = \overline{X}\,\overline{Y}$



Proof.

To prove this theorem, we need to recall complementarity laws, which state that

$$X + \overline{X} = 1 \text{ and } X.\overline{X} = 0$$

i.e., a logical variable/expression when added with its complement produces the output as 1 ... when multiplied with its complement produces the output as 0.

Now to prove DeMorgan's first theorem, we will use complementarity laws.

Let us assume that $P = X + Y$ where, P, X, Y are logical variables. Then, accordin... complementation law

$$P + \overline{P} = 1 \quad \text{and} \quad P.\overline{P} = 0.$$

That means, if P, X, Y are boolean variables then this complementarity law must hol... variable P. In other words, if $\overline{P}$ *i.e.*, if $\overline{X+Y} = \overline{X}\,\overline{Y}$ then

$(X+Y) + \overline{X}\,\overline{Y}$ must be equal to 1. (as X+...

and $(X+Y).\overline{X}\,\overline{Y}$ must be equal to 0. (as X....

Let us first prove the first part, *i.e.*,

$$(X+Y) + (\overline{X}\,\overline{Y}) = 1$$

$$(X+Y) + \overline{X}\,\overline{Y} = ((X+Y)+\overline{X}).((X+Y)+\overline{Y}) \quad \textit{(ref. } X + YZ = (X+Y)(X...$$

$$= (X + \overline{X} + Y).(X + Y + \overline{Y})$$

$$= (1 + Y).(X + 1) \quad \textit{(ref. } X+...$$

$$= 1.1 \quad \textit{(ref. } 1...$$

$$= 1$$

So first part is proved.

Now let us prove the second part *i.e.*,

$$(X+Y).\overline{X}\,\overline{Y} = 0$$

$$(X+Y).\overline{X}\,\overline{Y} = \overline{X}\,\overline{Y}.(X+Y) \quad \textit{(ref. } X(YZ)=...$$

$$= \overline{X}\,\overline{Y}\,X + \overline{X}\,\overline{Y}\,Y \quad \textit{(ref. } X(Y+Z)=X...$$

$$= X\,\overline{X}\,\overline{Y} + \overline{X}\,Y\,\overline{Y}$$

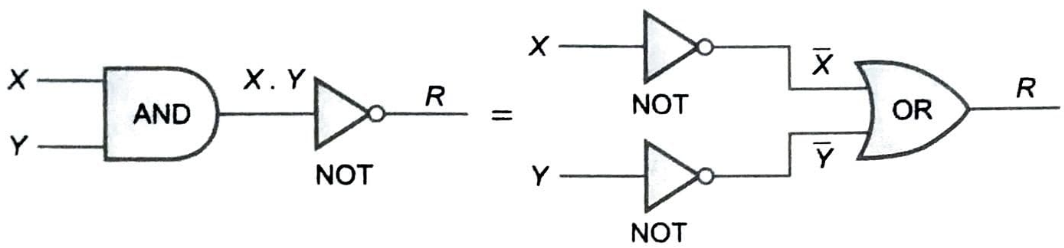
$$= 0.Y + \overline{X}.0 \quad \textit{(ref. } ...$$

$$= 0 + 0 = \mathbf{0}$$

So, second part is also proved, thus : $\overline{X+Y} = \overline{X}\,\overline{Y}$

1.10.2 DeMorgan's Second Theorem

This theorem states that : $\overline{X.Y} = \overline{X} + \overline{Y}$



Proof. Again to prove this theorem, we will make use of complementarity law *i.e.,*

$$X + \overline{X} = 1 \quad \text{and} \quad X . \overline{X} = 0.$$

If XY's complement is $\overline{X} + \overline{Y}$ then it must be true that

(*a*) $XY + (\overline{X} + \overline{Y}) = 1$ and (*b*) $XY(\overline{X} + \overline{Y}) = 0$

To prove the first part

$$\text{L.H.S} = XY + (\overline{X} + \overline{Y})$$

$$= (\overline{X} + \overline{Y}) + XY \qquad (\textit{ref.}\ X + Y = Y + X)$$

$$= (\overline{X} + \overline{Y} + X).(\overline{X} + \overline{Y} + Y) \qquad (\textit{ref.}\ (X + Y)(X + Z) = X + YZ)$$

$$= (X + \overline{X} + \overline{Y}).(\overline{X} + Y + \overline{Y})$$

$$= (1 + \overline{Y}).(\overline{X} + 1) \qquad (\textit{ref.}\ X + \overline{X} = 1)$$

$$= 1.1 \qquad (\textit{ref.}\ 1 + X = 1)$$

$$= 1 = \text{R.H.S}$$

Now the second part *i.e.,*

$$XY.(\overline{X} + \overline{Y}) = 0$$

$$\text{L.H.S} = XY.(\overline{X} + \overline{Y})$$

$$= XY\overline{X} + XY\overline{Y} \qquad (\textit{ref.}\ X(Y + Z) = XY + XZ)$$

$$= X\overline{X}Y + XY\overline{Y}$$

$$= 0.Y + X.0 \qquad (\textit{ref.}\ X.\overline{X} = 0)$$

$$= 0 + 0 = 0 = \text{R.H.S.}$$

$$XY.(\overline{X} + \overline{Y}) = 0 \quad \text{and} \quad XY + (\overline{X} + \overline{Y}) = 1$$

$\Rightarrow \overline{XY} = \overline{X} + \overline{Y}.$

Hence the theorem.

Although the identities above represent DeMorgan's theorem, the transformation is more easily performed by following these steps :

(*i*) Complement the entire function

(*ii*) Change all the ANDs (.) to ORs (+) and all the ORs (+) to ANDs (.)

(*iii*) Complement each of the individual variables.

1.11 DERIVATION OF BOOLEAN EXPRESSION

Boolean expressions which consist of a single variable or its complement *e.g.*, X or Y or $\overline{Z}$ are known as *literals*.

Now before starting derivation of boolean expression, first we will talk about two very important terms. These are (*i*) Minterms (*ii*) Maxterms

1.11.1 Minterms

One of the most powerful theorems within boolean algebra states that any boolean function can be expressed as the sum of products of all the variables within the system. For example, $X + Y$ can be expressed as the *sum of several products*, each of the product containing letters X and Y. These products are called *minterms* and each contains all the *literals* with or without the bar.

Definition

MINTERM is a product of all the literals (with or without the bar) within the logic system.

Also when values are given for different variables, minterm can easily be formed. *e.g.*, if $X = 0, Y = 1, Z = 0$ then minterm will be $\overline{X}Y\overline{Z}$ *i.e.*, for variable with a value 0, take its complement and the one with value 1, multiply it as it is. Similarly for $X = 1, Y = 0, Z = 0$, minterm will be $X\overline{Y}\,\overline{Z}$.

Steps involved in minterm expansion of expression

1. First convert the given expression in sum of products form.
2. In each term, if any variable is missing (*e.g.*, in the following example Y is missing *in first term* and X is missing in second term), multiply that term with (missing term + $\overline{\text{missing term}}$) factor, (*e.g.*, if Y is missing multiply with $Y + \overline{Y}$).
3. Expand the expression.
4. Remove all duplicate terms and we will have minterm form of an expression.

Example 1.20. *Convert $X + Y$ to minterms.*

Solution.

$$X + Y = X.1 + Y.1$$

$$= X.(Y + \overline{Y}) + Y(X + \overline{X}) \qquad (X + \overline{X} = 1 \text{ complementarity law})$$

$$= XY + X\overline{Y} + XY + \overline{X}Y$$

$$= XY + XY + X\overline{Y} + \overline{X}Y$$

$$= XY + X\overline{Y} + \overline{X}Y \qquad (XY + XY = XY \text{ Indempotent law})$$

Note that each term in the above example contains all the letters used : X and Y. The terms XY, $X\overline{Y}$ and $\overline{X}Y$ are therefore minterms. This process is called *expansion of expression*.

Other procedure for expansion could be

1. Write down all the terms
2. Put X's where letters much be inserted to convert the term to a product term
3. Use all combinations of X's in each term to generate minterms
4. Drop out duplicate terms.

Example 1.21. *Find the minterms for* $AB + C$.

Solution. It is a 3 variable expression, so a product term must have all three letters A, B and C

1. Write down all the terms $\quad AB + C$
2. Insert X's where letters are missing $\quad ABX + XXC$
3. Write all the combinations of X's in first term $\quad AB\overline{C}, ABC$

   Write all the combinations of X's in second term $\quad \overline{A}\,\overline{B}C, A\overline{B}C, ABC, \overline{A}BC$
4. Add all of them.

   Therefore, $AB + C = AB\overline{C} + ABC + \overline{A}\,\overline{B}C + A\overline{B}C + ABC + \overline{A}BC$
5. Now remove all duplicate terms

$$= AB\overline{C} + ABC + \overline{A}\,\overline{B}C + A\overline{B}C + \overline{A}BC$$

Now, to verify we will prove vice versa

i.e.,
$$AB\overline{C} + ABC + \overline{A}\,\overline{B}C + A\overline{B}C + \overline{A}BC = AB + C$$

$$\begin{aligned}
\text{L.H.S.} &= AB\overline{C} + ABC + \overline{A}\,\overline{B}C + A\overline{B}C + \overline{A}BC \\
&= \overline{A}\,\overline{B}C + \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC && \textit{(rearranging the terms)} \\
&= \overline{A}C(\overline{B} + B) + A\overline{B}C + AB(\overline{C} + C) \\
&= \overline{A}C.1 + A\overline{B}C + AB.1 && (\overline{B} + B = 1 \text{ and } \overline{C} + C = 1) \\
&= \overline{A}C + AB + A\overline{B}C \\
&= \overline{A}C + A(B + \overline{B}C) \\
&= \overline{A}C + A(B + C) && (X + \overline{X}Y = X + Y \textit{ Rule 18, Table 1.35}) \\
&= \overline{A}C + AB + AC \\
&= \overline{A}C + AC + AB \\
&= C(\overline{A} + A) + AB \\
&= C.1 + AB && (\overline{A} + A = 1) \\
&= C + AB \\
&= AB + C = \text{R.H.S.}
\end{aligned}$$

Shorthand minterm notation

Since all the letters (*2 in case of 2 variable expression, 3 in case of 3 variable expression*) must appear in every product, a shorthand notation has been developed that saves actually writing down the letters themselves. To form this notation, following steps are to be followed :

1. First of all, copy original terms.
2. Substitute 0's for barred letters and 1's for nonbarred letters.
3. Express the decimal equivalent of binary word as a subscript of m.

Example 1.22. *To find the minterm designation of* $X\overline{Y}\,\overline{Z}$.

Solution. 1. Copy Original form $= X\overline{Y}\,\overline{Z}$

2. Substitute 1's for non barred and 0's for barred letters

Binary equivalent $= 100$

Decimal equivalent of 100 $= 1 \times 2^2 + 0 \times 2^1 + 0 \times 2^0 = 4 + 0 + 0 = 4$

UNIT I

3. Express as decimal subscript of $m = m_4$

Thus $X\overline{Y}\,\overline{Z} = m_4$

Similarly, minterm designation of $A\overline{B}C\overline{D}$ would be

Copy Original Term $A\overline{B}C\overline{D}$

Binary equivalent $= 1\ 0\ 1\ 0$

Decimal equivalent $= 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 8 + 0 + 2 + 0 = 10$

Express as subscript of $m = m_{10}$

1.11.2 Maxterms

ISC XII

Trying to be logical about logic, if there is something called *minterm*, there surely must be one called *Maxterm* and there is.

Definition

A **MAXTERM** is a sum of all the literals (with or without the bar) within the logic system.

If the value of a variable is 1, then its complement is added otherwise the variable is added as it is. *e.g.*,

If the values of variables are $X = 0, Y = 1$ and $Z = 1$ then its Maxterm will be

$$X + \overline{Y} + \overline{Z}$$

(*Y and Z are 1's, so their complements are taken ; X = 0, so it is taken as it is*)

Similarly, if given values are $X = 1, Y = 0, Z = 0$ and $W = 1$ then its Maxterm is

$$\overline{X} + Y + Z + \overline{W}$$

Maxterms can also be written as M (capital M) with a subscript which is decimal equivalent of given input combination *e.g.*, above mentioned Maxterm $\overline{X} + Y + Z + \overline{W}$ whose input combination is 1001 can be written as M_9 as decimal equivalent of 1001 is 9.

1.11.3 Canonical Expression

Canonical expression can be represented in following *two* forms :

(*i*) Sum-of-Products (*S-O-P*) form (*ii*) Product-of-sums (*P-O-S*) form

Definition

Boolean Expression composed entirely either of minterms or maxterms is referred to as **CANONICAL EXPRESSION**.

Sum-of-Products (S-O-P)

A logical expression is derived from *two* sets of known values :

- various possible input values
- the desired output values for each of the input combinations.

Let us consider a specific problem.

A logical network has two inputs X and Y and an output Z. The relationship between inputs and outputs is to be as follows :

(*i*) When $X = 0$ and $Y = 0$ then $Z = 1$ (*ii*) When $X = 0$ and $Y = 1$ then $Z = 0$

(*iii*) When $X = 1$ and $Y = 0$, then $Z = 1$ (*iv*) When $X = 1$ and $Y = 1$, then $Z = 1$

We can prepare a truth table from the above relations which is as follows :

Table 1.36 *Truth table for product terms (2-input).*

X	Y	Z	Product Terms
0	0	1	$\overline{X}\,\overline{Y}$
0	1	0	$\overline{X}Y$
1	0	1	$X\overline{Y}$
1	1	1	XY

Here, we have added one more column to the table consisting list of *product terms* or *minterms*. Adding all the terms for which the output is 1 *i.e.*, $Z = 1$ we get following expression :

$$\overline{X}\,\overline{Y} + X\overline{Y} + XY = Z$$

Now see, it is an expression containing only minterms. This type of expression is called *minterm canonical form of boolean expression* or *canonical sum-of-products form of* expression.

Thus we can say

When a boolean expression is represented purely as sum of minterms, it is said to be in CANONICAL SUM-OF-PRODUCTS FORM.

Example 1.23. *A boolean function F defined on three input variables X, Y and Z is 1 if and only if number of 1 (one) inputs is odd (e.g., F is 1 if $X = 1, Y = 0, Z = 0$). Draw the truth table for the above function and express it in canonical sum-of-Products form.*

Solution. The output is 1, only if one of the inputs is odd. All the possible combinations when one of inputs is odd are

$$X = 1,\quad Y = 0,\quad Z = 0$$
$$X = 0,\quad Y = 1,\quad Z = 0$$
$$X = 0,\quad Y = 0,\quad Z = 1$$

for these combinations output is 1, otherwise output is 0. Preparing the truth table for it, we get the following truth table.

Table 1.37 *Truth table for product terms (3-input)*

X	Y	Z	F	Product Terms / Minterms
0	0	0	0	$\overline{X}\,\overline{Y}\,\overline{Z}$
0	0	1	1	$\overline{X}\,\overline{Y}Z$
0	1	0	1	$\overline{X}Y\overline{Z}$
0	1	1	0	$\overline{X}YZ$
1	0	0	1	$X\overline{Y}\,\overline{Z}$
1	0	1	0	$X\overline{Y}Z$
1	1	0	0	$XY\overline{Z}$
1	1	1	0	XYZ

Adding all the minterms (product terms) for which output is 1, we get

$$\overline{X}\,\overline{Y}Z + \overline{X}Y\overline{Z} + X\overline{Y}\,\overline{Z} = F$$

This is the desired *Canonical Sum-of-Products form.*

So, ***deriving S-O-P expression from Truth Table*** can be summarised as follows :

1. For a given expression, prepare a truth table for all possible combinations of inputs.
2. Add a new column for minterms and list the minterms for all the combinations.
3. Add all the minterms for which there is output as 1. This gives you the desired canonical *S-O-P* expression.

Another method of deriving canonical *S-O-P* expression is **Algebraic Method**. This is just the same as we have covered in section 1.11.1. We will take another example here.

Example 1.24. *Convert* $\overline{(\overline{X}Y) + (\overline{X}\,\overline{Z})}$ *into canonical sum of products form.*

Solution. Rule 1. Simplify the given expression using appropriate theorems/rules.

$$\overline{(\overline{X}Y) + (\overline{X}\,\overline{Z})} = (X + \overline{Y})(X + Z) \qquad \text{(using DeMorgan's laws)}$$

$$= X + \overline{Y}Z \qquad \text{(using Rule 15 of Table 1.35)}$$

Since it is a 3 variable expression, a product term must have all 3 variables

Rule 2. Wherever a literal is missing, multiply that term with (missing variable + $\overline{\text{missing variable}}$)

$$= X + \overline{Y}Z$$

$$= X(Y + \overline{Y})(Z + \overline{Z}) + (X + \overline{X})\overline{Y}Z$$

(Y, Z are missing in first term, X is missing in second term)

$$= (XY + X\overline{Y})(Z + \overline{Z}) + X\overline{Y}Z + \overline{X}\,\overline{Y}Z$$

$$= Z(XY + X\overline{Y}) + \overline{Z}(XY + X\overline{Y}) + X\overline{Y}Z + \overline{X}\,\overline{Y}Z$$

$$= XYZ + X\overline{Y}Z + XY\overline{Z} + X\overline{Y}\,\overline{Z} + X\overline{Y}\,\overline{Z} + \overline{X}\,\overline{Y}Z$$

Rule 3. Remove duplicate terms *i.e.,*

$$= XYZ + X\overline{Y}Z + XY\overline{Z} + X\overline{Y}\,\overline{Z} + \overline{X}\,\overline{Y}Z$$

This is the desired Canonical Sum-of-Products form.

Above Canonical Sum-of-Products expression can also be represented by following *shorthand notation e.g.,* $F = \Sigma(1, 4, 5, 6, 7)$ or $F = \sum m(1, 4, 5, 6, 7)$

where F is a variable function and m is a notation for minterm

This specifies that output F is sum of 1^{st}, 4^{th}, 5^{th}, 6^{th} and 7^{th} minterms *i.e.,*

$$F = m_1 + m_4 + m_5 + m_6 + m_7$$

Converting Shorthand Notation to Minterms

We already have learnt how to represent minterm into shorthand notation. Now we will learn how to convert vice versa.

Rule 1. Find binary equivalent of decimal subscript *e.g.,* for m_6 subscript is 6, binary equivalent of 6 is 110.

Rule 2. For every 1's write the variable as it is and for 0's write variable's complemented form *i.e.,* for 110 it is $XY\overline{Z}$, $XY\overline{Z}$ is the required minterm for m_6 .

Example 1.25. *Convert the following three input function F denoted by the expression :*

$F = \Sigma(0, 1, 2, 5)$ into its canonical Sum-of-Products form.

Solution. If three inputs we take as X, Y and Z then

$$F = m_0 + m_1 + m_2 + m_5$$

$$m_0 = 000 \Rightarrow \overline{X}\,\overline{Y}\,\overline{Z}$$

$$m_1 = 001 \Rightarrow \overline{X}\,\overline{Y}Z$$

$$m_2 = 010 \Rightarrow \overline{X}Y\overline{Z}$$

$$m_5 = 101 \Rightarrow X\overline{Y}Z$$

Canonical S-O-P form of the expression is

$$\overline{X}\,\overline{Y}\,\overline{Z} + \overline{X}\,\overline{Y}\,Z + \overline{X}\,Y\,\overline{Z} + X\,\overline{Y}\,Z$$

Product-of-Sum form

When a boolean expression is represented purely as product of Maxterms, it is said to be in canonical Product-of-Sum form of expression.

Note

When a boolean expression is represented purely as product of Maxterms, it is said to be in **CANONICAL PRODUCT-OF-SUM** form of expression.

This form of expression is also referred to as *Maxterm canonical form of boolean expression.*

Just as any boolean expression can be transformed into a sum of minterms, it can also be represented as a product of Maxterms.

(a) Truth Table Method

The truth table method for arriving at the desired expression is as follows :

1. Prepare a table of inputs and outputs
2. Add one additional column of sum terms. For each row of the table, a sum term is formed by adding all the variables in complemented or uncomplemented form *i.e.*, if input value for a given variable is 1, variable is complemented and if 0, not complemented. for $X = 0, Y = 1, Z = 1$, sum term will be $X + \overline{Y} + \overline{Z}$.

Now ***the desired expression is product of the sums from the rows in which the output is 0.***

Example 1.26. *Express in the product of sums form, the boolean function $F(x, y, z)$ and the truth table for which is given below :*

X	Y	Z	F
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

$= AB + \overline{A} + \overline{C} + A\overline{B}C$ (*putting* $\overline{AC} = \overline{A} + \overline{C}$ *DeMorgan's 2nd theorem*)

$= \overline{A} + AB + \overline{C} + A\overline{B}C$ (*rearranging the terms*)

$= \overline{A} + B + \overline{C} + A\overline{B}C$ (*putting* $\overline{A} + AB = A + B$ *because* $X + \overline{X}Y = X + Y$)

$= \overline{A} + \overline{C} + B + A\overline{B}C = \overline{A} + \overline{C} + B + \overline{B}AC$

$= \overline{A} + \overline{C} + B + AC$ (*putting* $B + \overline{B}AC = B + AC$ *because* $X + \overline{X}Y = X + Y$)

$= \overline{A} + B + \overline{C} + CA$

$= \overline{A} + B + \overline{C} + A$ ($\because \overline{C} + CA = \overline{C} + A$)

$= A + \overline{A} + B + \overline{C}$

$= 1 + B + \overline{C}$ (*putting* $A + \overline{A} = 1$)

$= 1$ (*as* $1 + X = 1$ *i.e., anything added to 1 results in 1*)

UNIT I

Example 1.32. *Reduce* $\overline{X}\,\overline{Y}\,\overline{Z} + \overline{X}Y\overline{Z} + X\overline{Y}\,\overline{Z} + XY\overline{Z}$.

Solution. $\overline{X}\,\overline{Y}\,\overline{Z} + \overline{X}Y\overline{Z} + X\overline{Y}\,\overline{Z} + XY\overline{Z} = \overline{X}(\overline{Y}\,\overline{Z} + Y\overline{Z}) + X(\overline{Y}\,\overline{Z} + Y\overline{Z})$

$= \overline{X}(\overline{Z}(\overline{Y} + Y)) + X(\overline{Z}(\overline{Y} + Y))$

$= \overline{X}(\overline{Z}.1) + X(\overline{Z}.1)$ ($\overline{Y} + Y = 1$)

$= \overline{X}\,\overline{Z} + X\overline{Z}$

$= \overline{Z}(\overline{X} + X)$

$= \overline{Z}.1$ ($\overline{X} + X = 1$)

$= \overline{Z}$

1.12.2 Simplification Using Karnaugh Maps

Truth Tables provide a nice, natural way to list all values of a function. There are several other ways to represent function values. One of them is *Karnaugh Map* (in short *K-Map*) named after its originator *Maurice Karnaugh*. These maps are sometimes also called *Veitch diagrams*.

What is Karnaugh Map ? *Karnaugh map or K-map is a graphical display of the fundamental products in a truth table.*

Karnaugh map is nothing but a rectangle made up of certain number of squares, each square representing a *Maxterm* or *Minterm*.

1.12.3 Sum-of-Products Reduction using Karnaugh Map

In *S-O-P* reduction each square of K-map represents a minterm of the given function. Thus, for a function of *n* variables, there would be a map of 2^n squares, each representing a minterm (refer to Fig. 1.7). Given a K-map, for *S-O-P* reduction the map is filled in by placing 1s in squares whose minterms lead to a 1 output.

Following are 2, 3, 4 variable K-maps for *S-O-P* reduction. (*see* Fig. 1.7)

Note in every square a number is written. These subscripted numbers denote that this square corresponds to that number's minterm. For example, in 3 variable map $\overline{X}\,Y\,\overline{Z}$ box has been given number 2 which means this square corresponds to m_2. Similarly, box number 7 means it corresponds to m_7 and so on.

Please notice the numbering scheme here, it is 0, 1, 3, 2 then 4, 5, 7, 6 and so on. Always squares are marked using this scheme while making a K-map.

X \ Y	[0] $\bar{Y}$	[1] Y
[0] $\bar{X}$	$\bar{X}\bar{Y}$	$\bar{X}Y$
[1] X	$X\bar{Y}$	XY

(a)

X \ Y	[0] $\bar{Y}$	[1] Y
[0] $\bar{X}$		
[1] X		

(b)

2-variable K-map representing minterms.

X \ YZ	[00] $\bar{Y}\bar{Z}$	[01] $\bar{Y}Z$	[11] YZ	[10] $Y\bar{Z}$
[0] $\bar{X}$	$\bar{X}\bar{Y}\bar{Z}$	$\bar{X}\bar{Y}Z$	$\bar{X}YZ$	$\bar{X}Y\bar{Z}$
[1] X	$X\bar{Y}\bar{Z}$	$X\bar{Y}Z$	XYZ	$XY\bar{Z}$

(c)

X \ YZ	[00] $\bar{Y}\bar{Z}$	[01] $\bar{Y}Z$	[11] YZ	[10] $Y\bar{Z}$
[0] $\bar{X}$				
[1] X				

(d)

3-variable K-map representing minterms

WX \ YZ	[00] $\bar{Y}\bar{Z}$	[01] $\bar{Y}Z$	[11] YZ	[10] $Y\bar{Z}$
[00] $\bar{W}\bar{X}$	$\bar{W}\bar{X}\bar{Y}\bar{Z}$	$\bar{W}\bar{X}\bar{Y}Z$	$\bar{W}\bar{X}YZ$	$\bar{W}\bar{X}Y\bar{Z}$
[01] $\bar{W}X$	$\bar{W}X\bar{Y}\bar{Z}$	$\bar{W}X\bar{Y}Z$	$\bar{W}XYZ$	$\bar{W}XY\bar{Z}$
[11] WX	$WX\bar{Y}\bar{Z}$	$WX\bar{Y}Z$	$WXYZ$	$WXY\bar{Z}$
[10] $W\bar{X}$	$W\bar{X}\bar{Y}\bar{Z}$	$W\bar{X}\bar{Y}Z$	$W\bar{X}YZ$	$W\bar{X}Y\bar{Z}$

(e)

WX \ YZ	[00] $\bar{Y}\bar{Z}$	[01] $\bar{Y}Z$	[11] YZ	[10] $Y\bar{Z}$
[00] $\bar{W}\bar{X}$				
[01] $\bar{W}X$				
[11] WX				
[10] $W\bar{X}$				

(f)

4-variable K-map representing minterms

FIGURE 1.7 2, 3, 4 variable K-maps for S-O-P expression

Observe carefully above given K-map. See the binary numbers at the top of K-map. These do not follow binary progression, instead they differ by only one place when moving from left to right:

00, 01, 11, 10. *It is done so that only one variable changes from complemented to uncomplemented form or vice versa.* **See** $\bar{A}\,\bar{B}$, $\bar{A}\,B$, AB, $A\bar{B}$.

This binary code 00, 01, 11, 10 is called **Gray Code**. *Gray Code is the binary code in which each successive number differs only in one place.* That is why box numbering scheme follows above order only.

How to map in K-map ?

We'll take an example of 2-variable map to illustrate this

Suppose, we have been given with the following truth table for mapping (Table 1.38).

Table. 1.38

A	B	F
0	0	0
0	1	0
1	0	1
1	1	1

Canonical *S-O-P* expression for this table is $F = A\overline{B} + AB$ or $F = \Sigma\,(2, 3)$.

To map this function first we'll draw an empty 2-variable K-map as shown in Fig. 1.8(*a*).

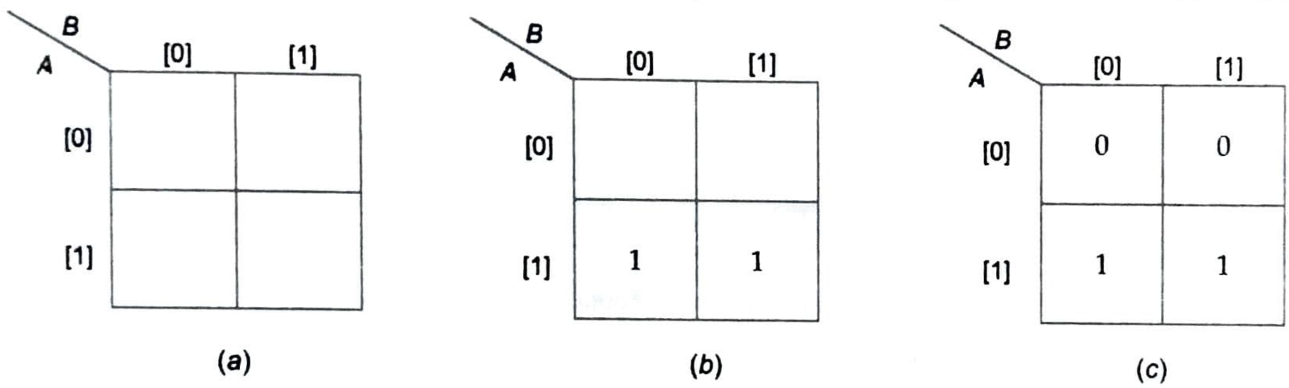


FIGURE 1.8 How to fill 2-variable K-map for a given truth table.

Now look for output 1 in the given truth table (1.38) for a given truth table.

For minterms m_2 and m_3 the output is 1. Thus mark 1 in the squares for m_2 and m_3 *i.e.*, square numbered as 2 and the one numbered as 3. Now our K-map will look like Fig. 1.8(*b*).

After entering 1's for all 1 outputs, enter 0's in all blank squares. K-map will now look like Fig 1.8(*c*). Same is the method for mapping 3-variable and 4-variable maps *i.e.*, enter 1's for all 1 outputs in the corresponding squares and then enter 0's in the rest of the squares.

How to reduce boolean expression in *S-O-P* form using K-map ?

For reducing the expression, first we have to mark *pairs, quads and octets.*

To reduce an expression, adjacent 1's are encircled. If *2 adjacent 1's* are encircled, it makes a *pair ; if 4 adjacent 1's* are encircled, it makes a *quad* ; and if *8 adjacent 1's* are encircled, it makes an *octet.*

While encircling groups of 1's, firstly search for *octets* and mark them, then for *quads* and lastly go for *pairs.* Thus is because a bigger group removes more variables thereby making the resultant expression simpler.

Reduction of a pair. In the K-map in Fig. 1.9, after mapping a given function $F\,(W, X, Y, Z)$ two pairs have been marked. Pair-1 is $m_0 + m_4$ (group of 0^{th} minterm and 4^{th} minterm as these numbers tell us minterm's subscript). Pair-2 is $m_{14} + m_{15}$.

Observe that Pair-1 is a vertical pair. Moving vertically in pair-1, see one variable X is changing its state from $\overline{X}$ to X as m_0 is $\overline{W}\ \overline{X}\ \overline{Y}\ \overline{Z}$ and m_4 is $\overline{W}\ X\ \overline{Y}\ \overline{Z}$. Compare the two and we see $\overline{W}\ \overline{X}\ \overline{Y}\ \overline{Z}$ changes to $\overline{W}\ X\ \overline{Y}\ \overline{Z}$. So, the variable X can be removed.

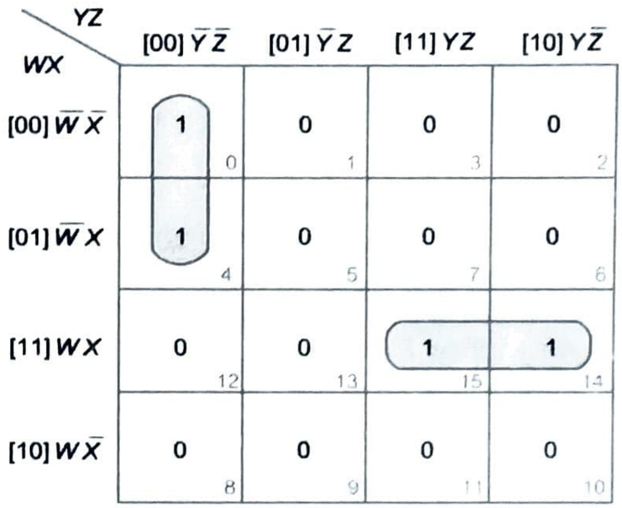


FIGURE 1.9 Pairs in a given K-map.

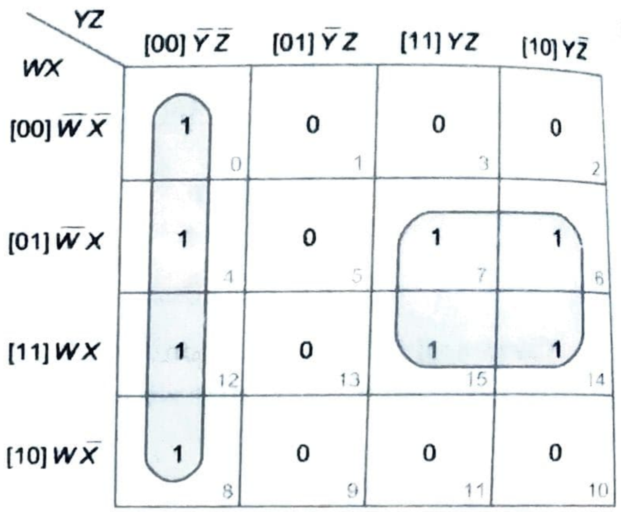


FIGURE 1.10 Quads in a given K-map

Pair Reduction Rule

Remove the variable which changes its state from complemented to uncomplemented or vice versa. Pair removes one variable only.

Thus reduced expression for Pair-1 is $\overline{W}\ \overline{Y}\ \overline{Z}$ as $\overline{W}\ \overline{X}\ \overline{Y}\ \overline{Z}\ (m_0)$ changes to $\overline{W}\ X\ \overline{Y}\ \overline{Z}\ (m_4)$.

We can prove the same algebraically also as follows :

$$\begin{aligned}\text{Pair-1} &= m_0 + m_4 = \overline{W}\ \overline{X}\ \overline{Y}\ \overline{Z} + \overline{W}\ X\ \overline{Y}\ \overline{Z}\\ &= \overline{W}\ \overline{Y}\ \overline{Z}\ (\overline{X} + X)\\ &= \overline{W}\ \overline{Y}\ \overline{Z}\ .1 \qquad (\overline{X} + X =\\ &= \overline{W}\ \overline{Y}\ \overline{Z}\end{aligned}$$

Similarly, reduced expression for Pair-2 $(m_{14} + m_{15})$ will be WXY as $W\ X\ Y\ Z\ (m_{14})$ changes t $WXY\overline{Z}\ (m_{15})$. Z will be removed as it is changing its state from $\overline{Z}$ to Z.

Reduction of a quad

If we are given with the K-map shown in Fig. 1.10 in which two quads have been marked.

Quad-1 is $m_0 + m_4 + m_{12} + m_8$ and Quad-2 is $m_7 + m_6 + m_{15} + m_{14}$. When we move across quad-1, two variables change their states *i.e.*, W and X are changing their states, so these tw variables will be removed.

Quad. Reduction Rule

Remove the two variables which change their states. A Quad removes two variables. Thus reduce expression for Quad-1 is $\overline{Y}\ \overline{Z}$ as W and X (both) are removed.

Similarly, in Quad-2 $(m_7 + m_6 + m_{15} + m_{14})$, horizontally moving, variable Z is removed a $\overline{W}\ X\ Y\ Z\ (m_7)$ changes to $\overline{W}\ X\ Y\ \overline{Z}\ (m_6)$ and vertically moving, variable W is removed as $\overline{W}\ X\ Y\ Z$(m changes to $WXYZ$. Thus reduced expression for quad-2 is (by removing W and Z) XY.

Reduction of an octet

Suppose, we have K-map with an octet marked as shown in Fig. 1.11.

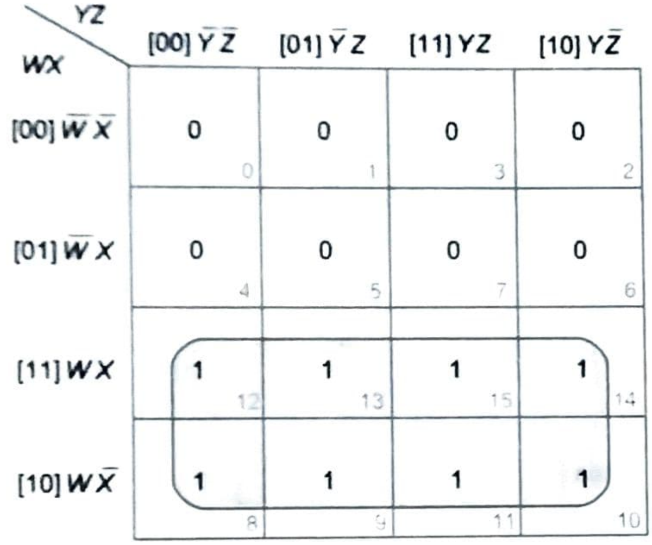


FIGURE 1.11 Octets in a given K-map.

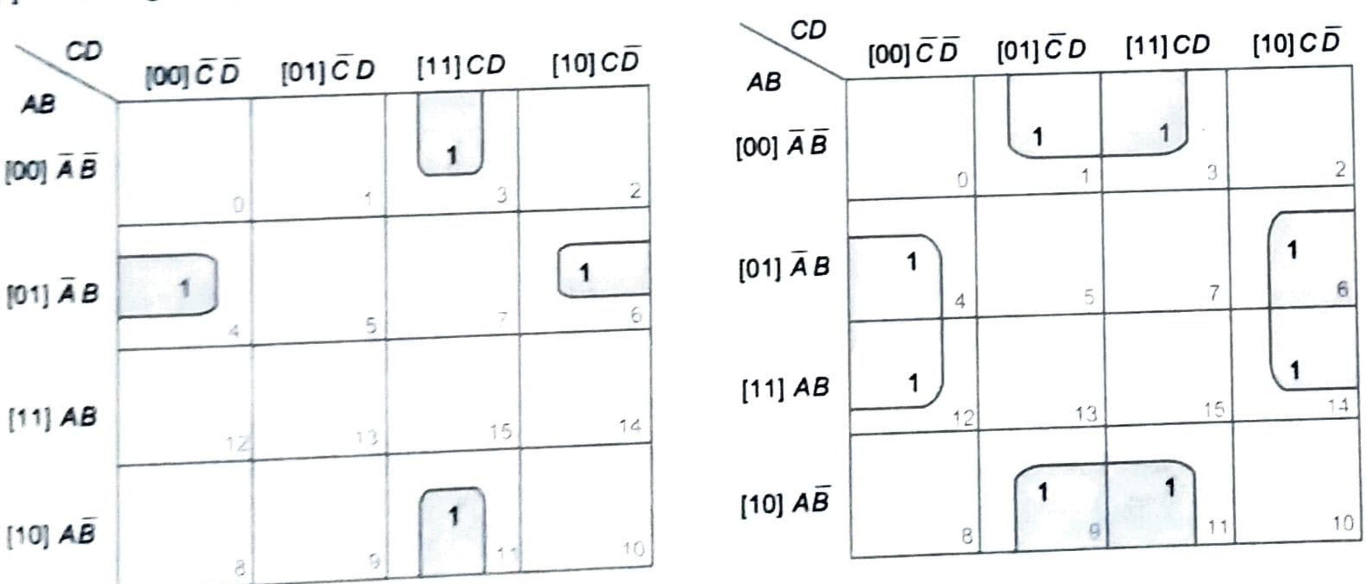
While moving horizontally in the octet two variables Y and Z are removed and moving vertically one variable X is removed. Thus eliminating X, Y and Z, the reduced expression for the octet is W only.

Octet Reduction Rule

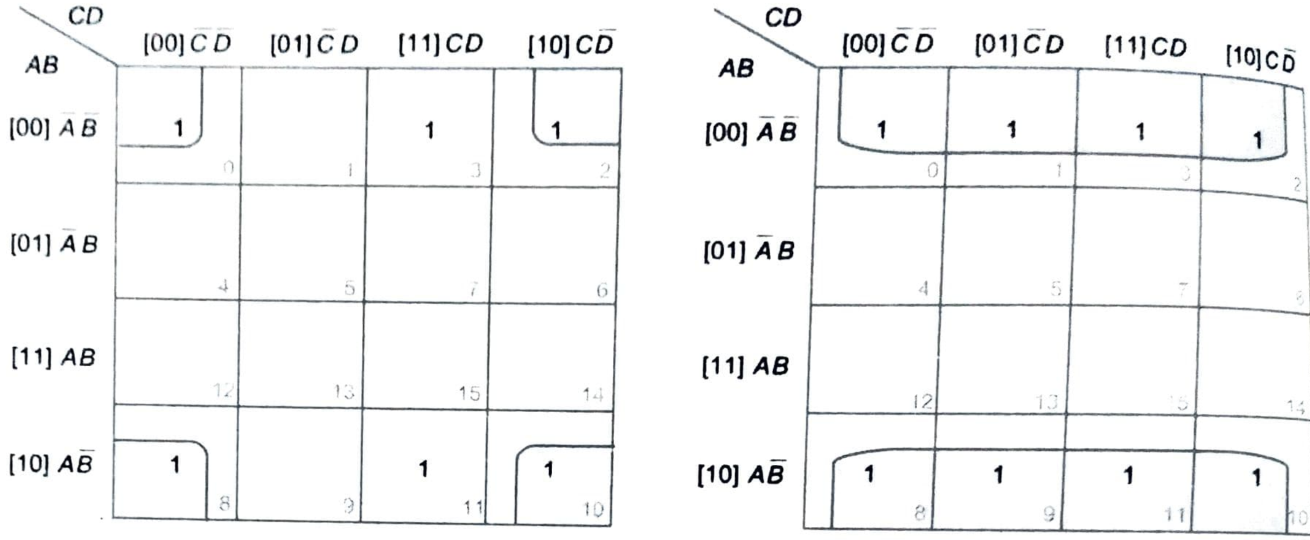
Remove the three variables which change their states. An octet removes 3-variables. But after marking pairs, quads and octets, there are certain other things to be taken care of before arriving at the final expression. These are map rolling, overlapping groups and reduntant groups.

Map Rolling

Map Rolling means roll the map *i.e., consider the map as if its left edges are touching the right edges and top edges are touching bottom edges.* This is a special property of Karnaugh maps that its opposite edges squares and corner squares are considered contiguous (Just as the world map is



(*a*) Pairs (*b*) Quads



(c) quad (b) octet

FIGURE 1.12 Map rolling.

treated contiguous at its opposite ends). As *in opposite edges squares and in corner squares only o* *variable changes its state from complemented to uncomplemented state or vice versa*. Therefore, whi marking the pairs, quads and octets, map must be rolled. Following pairs, quads and octets a marked after rolling the map.

Overlapping Groups

Overlapping means same 1 can be encircled more than once. For example, if the following K-ma (Fig. 1.13) is given :

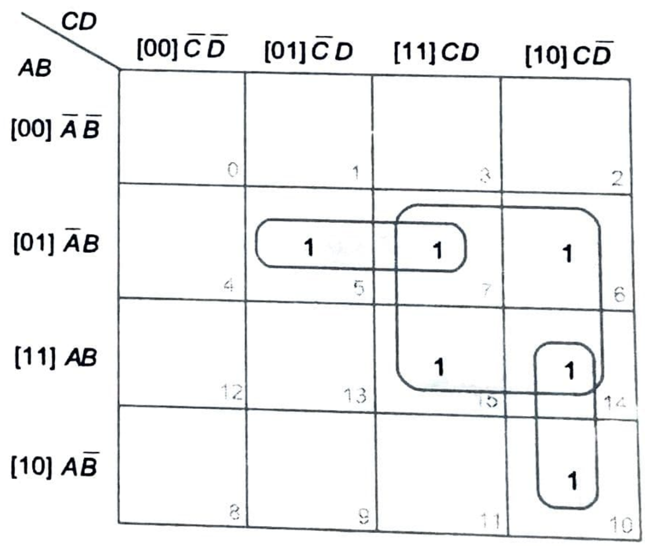


FIGURE 1.13 Overlapping Groups.

Observe that 1 for m_7 has been encircled twice. Once for Pair-1 ($m_5 + m_7$) and again for Qua ($m_7 + m_6 + m_{15} + m_{14}$). Also 1 for m_{14} has been encircled twice. For the Quad and for Pair-2 ($m_{14} + m_{10}$)

Overlapping always leads to simpler expressions.

Here, reduced expression for Pair-1 is $\bar{A}BD$

reduced expression for Quad is BC

reduced expression for Pair-2 is $AC\bar{D}$

Thus final reduced expression for this map (Fig. 1.13) is

$$\bar{A}BD + BC + AC\bar{D}$$

Thus *reduced expression for entire K-map is sum of all reduced expressions in the very K-map.*

But before writing the final expression we must take care of ***Reduntant Groups.***

Reduntant Group

Reduntant Group *is a group whose all 1's are overlapped by other groups (i.e., pairs, quads, octets).* Here is an example, given below in Fig. 1.14.

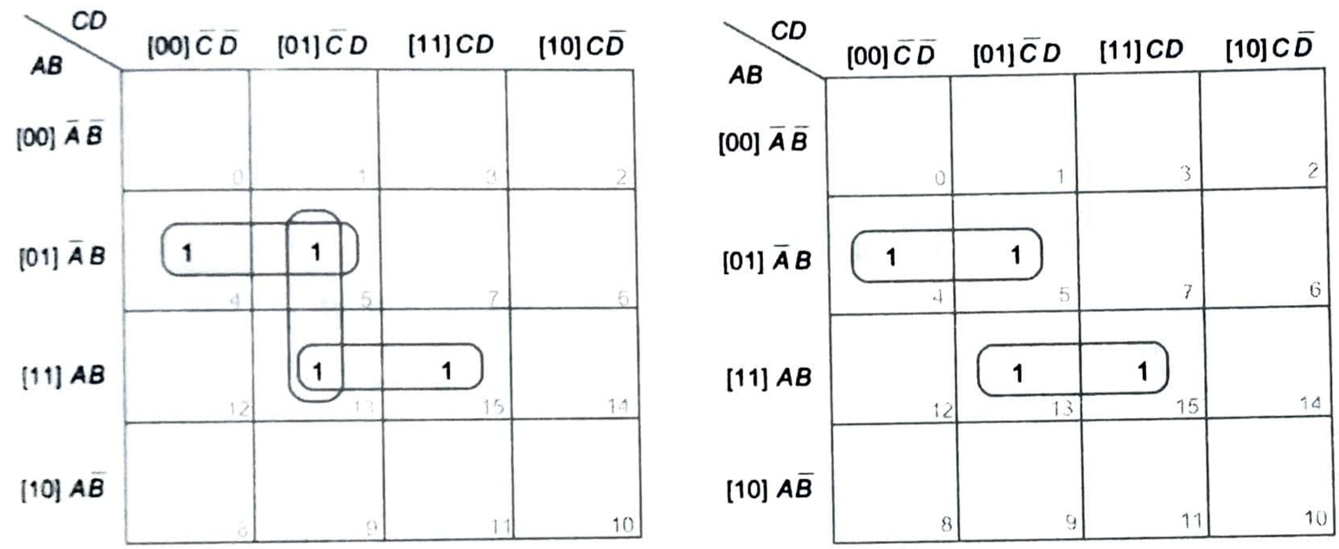


FIGURE 1.14 (a) K-map with reduntant group (b) K-map without reduntant group.

Fig. 1.14(*a*) has a reduntant group. There are three pairs : Pair-1 ($m_4 + m_5$), Pair-2 ($m_5 + m_{13}$), Pair-3 ($m_{13} + m_{15}$). Bur Pair 2 is a reduntant group as its all 1's are marked by other groups.

With this reduntant group, the reduced expression will be $\bar{A}B\bar{C} + B\bar{C}D + ABD$. For a simpler expression, ***Reduntant Groups must be removed***. After removing the reduntant group, we get the K-map shown in Fig. 1.14(*b*).

The reduced expression, for K-map in Fig. 1.14(*b*), will be

$$\bar{A}B\bar{C} + ABD$$

which is much simpler expression.

Thus ***removal of reduntant group leads to much simpler expression.***

Summary of all the rules for S-O-P reduction using K-map

1. Prepare the truth table for given function.
2. Draw an empty K-map for the given function (*i.e.*, 2 variable K-map for 2 variable function ; 3 variable K-map for 3 variable function, and so on).
3. Map the given function by entering 1's for the outputs as 1 in the corresponding squares.

4. Enter 0's in all left out empty squares.
5. Encircle adjacent 1's in form of *octets, quads* and *pairs*. Do not forget to roll the map and overlap.
6. Remove reduntant groups, if any.
7. Write the reduced expressions for all the groups and OR (+) them.

Example 1.33. *Reduce* $F\,(a,\,b,\,c,\,d) = \Sigma\, m\,(0,\,2,\,7,\,8,\,10,\,15)$ *using Karnaugh map.*

Solution. Given $F\,(a,b,c,d) = \Sigma m(0,2,7,8,10,15)$

$$= m_0 + m_2 + m_7 + m_8 + m_{10} + m_{15}$$

$m_0 = 0000 = \overline{A}\ \overline{B}\ \overline{C}\ \overline{D}$ $\quad m_2 = 0010 = \overline{A}\ \overline{B}\ C\ \overline{D}$

$m_7 = 0111 = \overline{A}BCD$ $\quad m_8 = 1000 = A\overline{B}\ \overline{C}\ \overline{D}$

$m_{10} = 1010 = A\overline{B}C\overline{D}$ $\quad m_{15} = 1111 = ABCD$

Truth Table for the given function is as follows :

A	B	C	D	F
0	0	0	0	1
0	0	0	1	
0	0	1	0	1
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	1
1	0	0	0	1
1	0	0	1	
1	0	1	0	1
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	1

Mapping the given function in a K-map, we get

AB \ CD	[00] $\overline{C}\,\overline{D}$	[01] $\overline{C}D$	[11] CD	[10] $C\overline{D}$
[00] $\overline{A}\,\overline{B}$	1 (0)	0 (1)	0 (3)	1 (2)
[01] $\overline{A}B$	0 (4)	0 (5)	1 (7)	0 (6)
[11] AB	0 (12)	0 (13)	1 (15)	0 (14)
[10] $A\overline{B}$	1 (8)	0 (9)	0 (11)	1 (10)

In the above K-map two groups have been marked, one Pair and One Quad.

Pair is $m_7 + m_{15}$

and Quad is $m_0 + m_2 + m_8 + m_{10}$

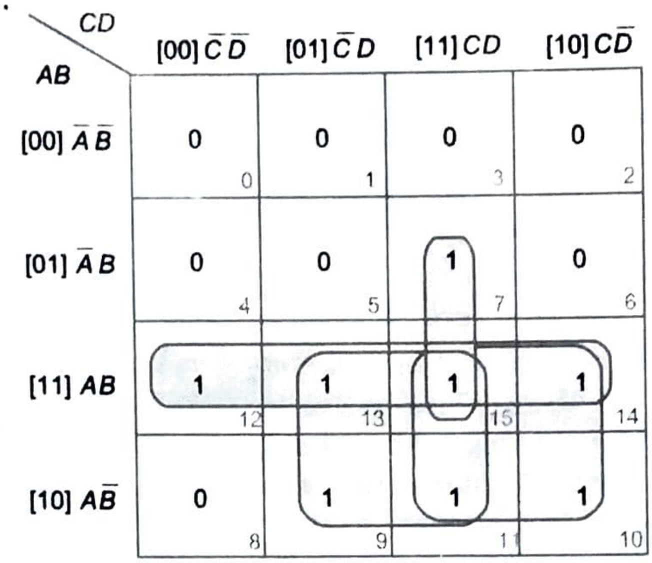
Reduced expression for pair $(m_7 + m_{15})$ is ***BCD*** as *A* is removed. Reduced expression for quad $(m_0 + m_2 + m_8 + m_{10})$ is $\boldsymbol{\overline{B}\,\overline{D}}$ as for horizontal corners C is removed and for vertical corners A is removed.

Thus final reduced expression is $BCD + \overline{B}\,\overline{D}$.

Example 1.34. *What is the simplified boolean equation for the function :*

$$F(A, B, C, D) = \Sigma(7, 9, 10, 11, 12, 13, 14, 15)\ ?$$

Solution. *Completing the given Karnaugh map by entering 0's in the empty squares, by numbering the squares with their minterm's subscripts and the:: by encircling all possible groups,* we get the following K-map.



There is one pair, three *quads*

Pair-1 $= m_7 + m_{15}$;

Quad-1 $= m_{12} + m_{13} + m_{15} + m_{14}$

Quad-2 $= m_{13} + m_{15} + m_9 + m_{11}$;

Quad-3 $= m_{15} + m_{11} + m_{14} + m_{10}$

Reduced expression for pair-1 $(m_7 + m_{15})$ is **BCD**, as $\bar{A}BCD(m_7)$ changes to ABCD (m_{15}) eliminating *A*.

Reduced expression for Quad-1 $(m_{12} + m_{13} + m_{15} + m_{14})$ is ***AB***, as while moving across the Quad, *C* and *D* both are removed because both are changing their states from complemented to uncomplemented or vice-versa.

Reduced expression for Quad 2 $(m_{13} + m_{15} + m_9 + m_{11})$ is ***AD***, as moving horizontally, *C* is removed and moving vertically, *B* is removed.

Reduced expression of Quad-3 $(m_{15} + m_{11} + m_{14} + m_{10})$ is ***AC*** as horizontal movement removes *D* and vertical movement removes *B*.

Thus, Pair-1 $= BCD$, Quad-1 $= AB$, Quad-2 $= AD$, Quad-3 $= AC$

Hence finai reduced expression will be ***BCD* + *AB* + *AD* + *AC*.**

Example 1.35. *Obtain a simplified expression for a Boolean function F (X, Y, Z), the Karnaugh map for which is given below :*

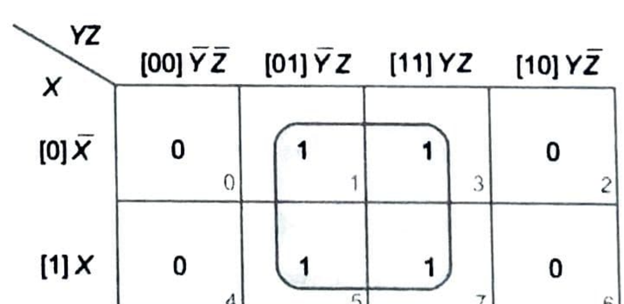
X \ Y	[00]	[01]	[11]	[10]
[0]		1	1	
[1]		1	1	

Solution.

Completing the given K-map,

We have 1 group which is a Quad *i.e.*,

$$m_1 + m_3 + m_5 + m_7$$



Reduced expression for this Quad is Z, as moving horizontally from $\bar{X}\,\bar{Y}Z(m_1)$ to $\bar{X}YZ(m_3)$, Y is removed (Y changing from $\bar{Y}$ to Y) and moving vertically from m_1 to m_5 or m_3 to m_7, $\bar{X}$ changes to X, thus *X* is removed.

Example 1.36. *Minimise the following function using a Karnaugh map:*

$$F(W, X, Y, Z) = \Sigma(0, 4, 8, 12).$$

Solution. Given function $F(W, X, Y, Z) = \Sigma(0, 4, 8, 12)$

$$F = m_0 + m_4 + m_8 + m_{12}$$

$m_0 = 0000 = \overline{W}\,\overline{X}\,\overline{Y}\,\overline{Z}$,

$m_4 = 0100 = \overline{W}X\overline{Y}\,\overline{Z}$,

$m_8 = 1000 = W\overline{X}\,\overline{Y}\,\overline{Z}$,

$m_{12} = 1100 = WX\overline{Y}\,\overline{Z}$

WX \ YZ	[00] $\overline{Y}\,\overline{Z}$	[01] $\overline{Y}Z$	[11] YZ	[10] $Y\overline{Z}$
[00] $\overline{W}\,\overline{X}$	1	0	0	0
[01] $\overline{W}X$	1	0	0	0
[11] WX	1	0	0	0
[10] $W\overline{X}$	1	0	0	0

Mapping the given function on a K-map, we get

Only 1 group is here, a Quad

$$(m_0 + m_4 + m_{12} + m_8)$$

Reduced expression for this quad is $\overline{Y}\,\overline{Z}$, as while moving across the Quad W and X are removed because these are changing their states from complemented to uncomplemented or vice versa.

Thus, final reduced expression is $\overline{Y}\,\overline{Z}$.

Example 1.37. *Using the Karnaugh technique obtain the simplified expression as sum of products for the following map :*

X \ YZ	[00]	[01]	[11]	[10]
[0]			1	1
[1]			1	1

Solution. Completing the given K-map, we get

One group which is a Quad has been marked.

X \ YZ	[00] $\overline{Y}\,\overline{Z}$	[01] $\overline{Y}Z$	[11] YZ	[10] $Y\overline{Z}$
[0] $\overline{X}$	0	0	1	1
[1] X	0	0	1	1

Quad reduces *two variables*. Moving horizontally, Z is removed as it changes from Z to $\overline{Z}$ and moving vertically, X is removed as it changes from $\overline{X}$ to X. Thus only one variable Y is left. Hence Reduced *S-O-P* expression is Y. Thus $F = Y$ assuming F is the given function.

1.12.4 Product-of-Sum Reduction using Karnaugh Map

In *P-O-S* reduction each square of K-map represents a Maxterm. Karnaugh map is just the same as that of the used in *S-O-P* reduction. For a function of n variables, map would represent 2^n squares, each representing a Maxterm.

For *P-O-S* reduction map is filled by placing 0's in squares whose Maxterms lead to output 0. Following are 2, 3, 4 variable K-maps for *P-O-S* reduction.

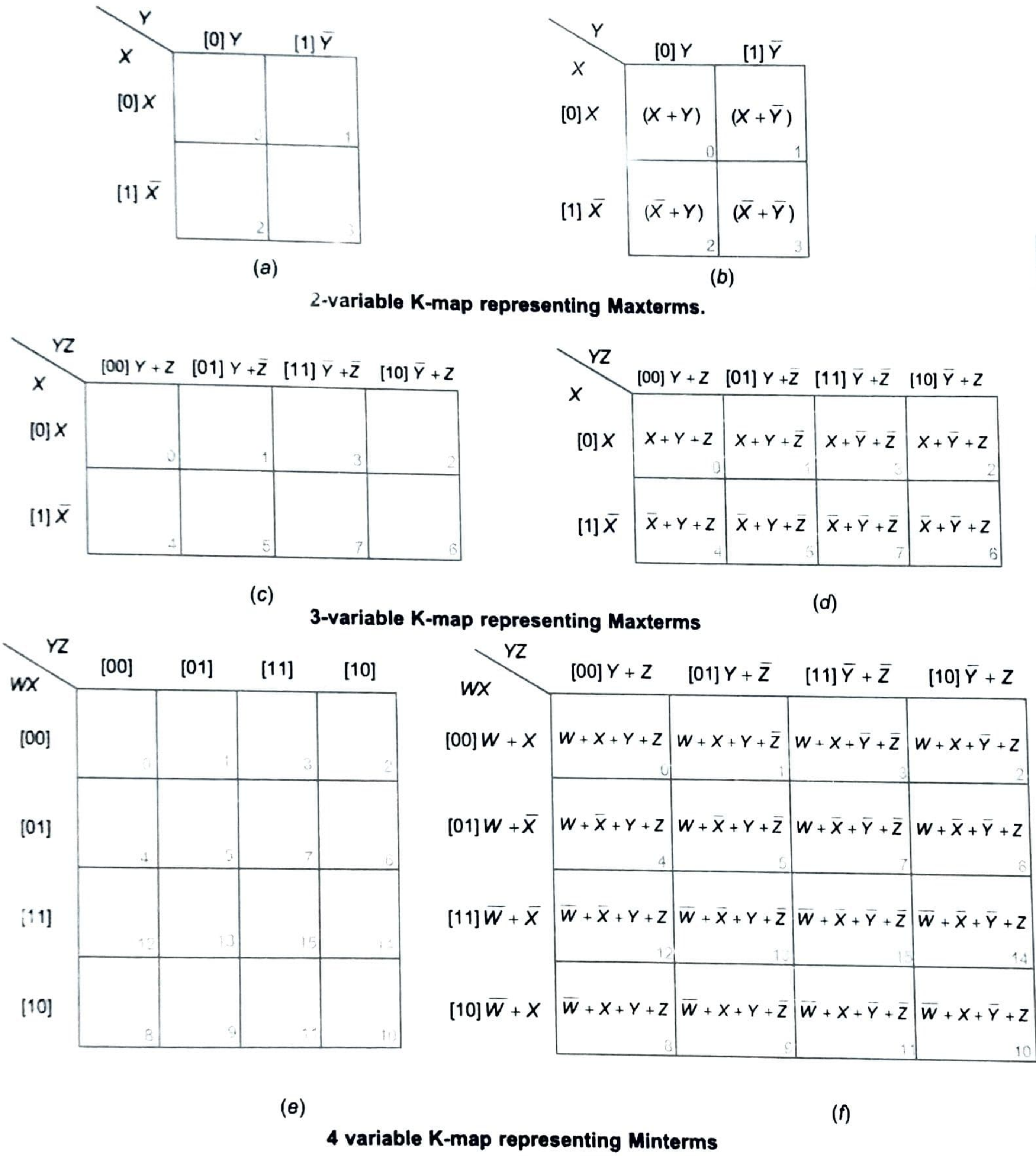


FIGURE 1.15 2,3,4 variable K-Maps of P-O-S expression.

Again, the numbers in the squares represent *Maxterm subscripts*. Box with number 1 represents M_1, number 6 box represents M_6, and so on. Also notice *box numbering scheme is the same i.e.*, 0, 1, 3, 2 ; 4, 5, 7, 6 ; 12, 13, 15, 14 ; 8, 9, 11, 10.

One more similarity in *S-O-P* K-map and *P-O-S* K-map is that they are binary progression in Gray code only. So, here also same Gray Code appears at the top.

But one major difference is that in P-O-S K-map, complemented letters represent 1's and uncomplemented letters represent 0's whereas it is just the opposite in S-O-P K-map. Thus in the Fig. 1.15 (*b*), (*d*), (*f*) for 0's uncomplemented letters appear and for 1's complemented letters appear.

How to derive P-O-S boolean expression using K-map

Rules for deriving expression are the same except for one thing *i.e., for P-O-S expression adjacent 0's are encircled in the form of pairs, quads and octets.* Therefore, rules for deriving P-O-S boolean expression can be summarized as follows :

ISC XII

1. Prepare the truth table for a given function.
2. Draw an empty K-map for given function (*i.e.,* 2-variable K-map for 2-variable function, 3-variable K-map for 3-variable function and so on).
3. Map the given function by entering 0's for the outputs as 0 in the corresponding squares. (*i.e.,* if M_5 and M_{13} are 0's then squares numbered 5 and 13 will be having 0's).
4. Enter 1's in all left out empty squares.
5. Encircle adjacent 0's in the form of octets, quads, and pair. Do not forget to roll the map and overlap.
6. Remove reduntant groups if any.
7. Write the reduced expressions for all the groups and AND (.) them.

Example 1.38. *Reduce the following Karnaugh map in Product-of-sums form:*

A \ BC	[00]	[01]	[11]	[10]
[0]	0	0	0	1
[1]	0	1	1	1

Solution. To reach at *P-O-S* expression, we'll have to encircle all possible groups of adjacent 0's. Encircling we get the following K-map.

A \ BC	[00] $B + C$	[01] $B + \overline{C}$	[11] $\overline{B} + \overline{C}$	[10] $\overline{B} + C$
[0] A	0	0	0	1
[1] $\overline{A}$	0	1	1	1

There are 3 pairs which are :

Pair-1 = $M_0 \cdot M_1$;

Pair-2 = $M_0 \cdot M_4$;

Pair-3 = $M_1 \cdot M_3$.

But there is one reduntant group also *i.e.,* Pair-1 (its all 0's are encircled by other groups). Thus removing this reduntant pair-1, we have only two groups now.

Reduced *P-O-S* expression for Pair-2 is **($B + C$)**, as while moving across pair-2, A changes its state from A to $\overline{A}$, thus A is removed.

Reduced P-O-S expression for Pair-3 is $(A + \bar{C})$, as while moving across Pair-3 B changes to $\bar{B}$, hence eliminated.

Final P-O-S expression will be

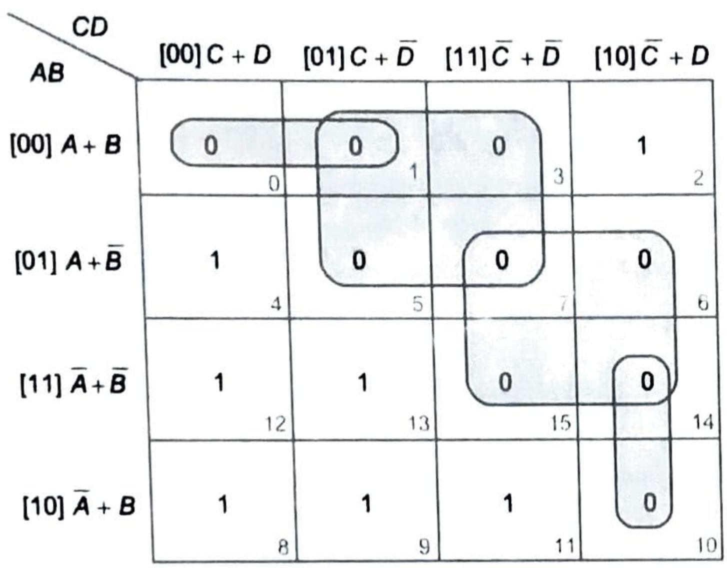
$$(B + C).(A + \bar{C})$$

Example 1.39. *Find the minimum P-O-S expression of*

$$Y(A, B, C, D) = \prod(0, 1, 3, 5, 6, 7, 10, 14, 15).$$

Solution. As the given function is 4-variable function, we'll draw 4-variable K-map and then put 0's for the given Maxterms *i.e.*, in the squares whose numbers are 0, 1, 3, 5, 6, 7, 10, 14, 15 as each square number represents its Maxterm.

So, K-map will be



Encircling adjacent 0's we have following groups :

Pair-1 $= M_0 \cdot M_1$; Pair-2 $= M_{14} \cdot M_{10}$;

Quad-1 $= M_1 \cdot M_3 \cdot M_5 \cdot M_7$; Quad-2 $= M_7 \cdot M_6 \cdot M_{15} \cdot M_{14}$

Reduced expressions are as follows :

For Pair-1, $(A + B + C)$ (*as D is eliminated : D changes to* $\bar{D}$)

For Pair-2, $(\bar{A} + \bar{C} + D)$ ($\bar{B}$ *changes to B ; hence eliminated*)

For Quad-1, $(A + \bar{D})$ (horizontally C and vertically B is eliminated as C, B are changing their states)

For Quad-2, $(\bar{B} + \bar{C})$ (horizontally D and vertically A is eliminated)

Hence final P-O-S expression will be

$$Y(A, B, C, D) = (A + B + C)\,(\bar{A} + \bar{C} + D)\,(A + \bar{D})\,(\bar{B} + \bar{C})$$